Spring 3-1-2002

Assessment Office Online Employer Survey

Aluisius Sudiarto
Dakota State University

DAKOTA STATE UNIVERSITY

MSIS

PROJECT APPROVAL FORM

Student Name: Aluisius Sudiarto

Expected Graduation Date: 12/15/2002

Master's Project Title: Assessment Office Online Employer Survey

Date Project Plan Approved: 03/27/2002

Date Project Coordinator Notified and Grade Submitted: 12/2001

Approvals/Signatures:

Student: _____ Date: 1-23-03

Faculty supervisor: _____ Date: 1-23-03

Committee member: _____ Date: 1-23-03

Committee member: _____ Date: 1-23-03

M.S.I.S. Graduate Project

Web-Based Employer Survey

Dakota State University – Assessment Office

By Aluisius Sudiarto

A project submitted in partial fulfillment of the requirements for the

Master of Science in Information Systems

Dakota State University

Year 2002

Abstract

The project was one of my tasks while I was working for DSU's Assessment Office. The project's goal was to create a web-based employer survey from the paper-based employer survey. The project used Active Server Pages (ASP), Hyper Text Mark-up Language (HTML), Cascading Style Sheet (CSS), JavaScript, and Microsoft Access. Microsoft Access was used because it was the requirement from the Assessment Office.

The survey is under the responsibility of the Assessment Office. It has been used for 9 years. The project introduced the survey through the Internet for the first time.

This report explains the method and the technologies used to build the survey. There are some code samples and figures to help understand how the actual codes work. The report also explains the database connectivity with ASP.

ASP was used to make the connection between the HTML pages and the Microsoft Access database. This connection includes, but is not limited to, adding, editing, deleting, and updating the database.

HTML pages were used for presentation of the survey. To make the pages look uniform, CSS was used. The font type and font size for the survey questions and for the form header are regulated with CSS.

JavaScript was used to validate user entries. To minimize the number of unanswered questions and to prevent duplicate entry, the input had to be validated before it could be submitted.

Table of Contents

Appendices

List of Figures

List of Tables

Introduction

This report documents the steps that were necessary to complete the project of converting a paper and pencil employer survey to a web based survey. It could be used as a supplemental guide to do a similar project.

The project was part of my work as a graduate assistant in Dakota State University's Assessment Office. Carrie Ahern, DSU Assessment Specialist, requested that I develop an internet-based survey as part of my work. This task became my graduate project in the Master of Science in Information System program. The Dakota State University graduate faculty approved my proposal and it officially became a graduate project.

The project's goal was to make an internet based survey from the previously paper-based one. The survey was an employer survey that had been used for 9 years by the Assessment Office. The information gathered from the survey had been used to determine if the Dakota State University (DSU) goals and outcomes were met by the general education and major field assessment plans.

The project used Hyper Text Mark-up Language (HTML) to create the survey forms, Cascading Style Sheet to maintain the style and layout of the form, JavaScript to validate data and to check data input error, Microsoft Access as the database (system requirement from the Assessment Office), and Active Server Pages (ASP) to connect the

survey forms with the database and to control the page flow. The web pages and the database were stored on the Assessment Office space in the DSU's web server.

Employers, in this case are companies or organizations, evaluate DSU's graduates based on their previous year's work performance. The employers rate the graduates in three areas: computer skills, communication/socialization skills, and job related skills.

There were two types of input for the survey questions. One type of input was options. In this input type, questions could be scaleable. On a scale of 1 to 5, one was the lowest and five was the highest, employers chose one option as an input for related questions. Other questions in this input type were frequency questions. These questions were using options ranging from significantly less than usual to significantly more than usual. Employers can only choose one answer for each question. The other input type was text. In this type of input, the survey asked the employers for comments or suggestions. Employers inputted comments or suggestions through a Hyper Text Mark-up Language (HTML) form textbox.

The respondent for the survey could be categorized into two groups: education major graduates and non-education major graduates. For education major graduates, employers would get another set of survey questions. This requirement came from the Assessment Office.

All of the survey results would be saved in a Microsoft Access database. The database consisted of four tables. There were two tables that stored the information about the company and the employee and another two tables that stored the survey result, one for education graduates and one for non-education graduates.

Problems and Questions Encountered

The first question from Carrie Ahern about this project was confidentiality. Employers wanted an assurance of the secrecy of the survey results. The solution for this issue was to create a password protected survey. Only a person with the right information, company identification and password, could access his or her part of the survey.

By using Active Server Pages (ASP), the survey also is more secure. Buser, et al. (2000) stated "The fact that the browser doesn't allow the end-user to view the ASP source" (p. 53).

Another issue that came up during the planning process was the internet browser compatibility. All of the database connections are coded with ASP. According to Buser, et al. (2000) "ASP is server-side scripting" (p. 66) and one of the advantage of server-side scripting is that it enables you to program dynamic web applications browser-independently (p. 19).

I encountered some other problems during the project. I will explain more of those problems later in this report.

Project Objectives

The main objective of the project was to transform the paper-based survey, which had been used for nine years, to an online or internet based survey.

There should be significant improvements by transforming the survey to the web based survey. According to www.fcw.com:

> "NREL's first online survey, conducted last year, had a response rate of 54 percent, well above the expected rate of 33 percent and easily outpacing paper-based surveys conducted in previous years, said Bryan Mohler, director of quality and assessment at NREL."[1]

In addition to that, according to www.websurveyor.com:

> "...the purpose of the survey and the notification method used, responses can start to accumulate in minutes - complete surveys can be conducted in a few days." But, "...paper-based surveys take from several weeks to several months to distribute, collect and encode for analysis."

And,

> "Web surveys provide better control surfaces than do paper surveys. While radio buttons (select one) and check boxes (check all that apply) are common to both, web surveys can also take advantage of pull down lists."

And also,

> "Asking people to provide text based subjective feedback is another area where web surveys have a distinct advantage ... In web surveys it is very easy to add multiple verbose text areas that the user can fill in."[2]

---

[1] http://www.fcw.com/fcw/articles/2001/0903/tec-doe-09-03-01.asp

[2] http://www.websurveyor.com/learn_news0110_bp.asp

By the end of the project, the survey forms and a database that contained all the survey results were created. Survey questions displayed in a HTML page and survey results were stored into a Microsoft Access file.

Employers would need two credentials to access the survey: company ID and password. These credentials were sent to employers through regular mail or through e-mail. Along with the credentials, a hyperlink to the survey's login page was also provided. The link would be a direct hyperlink in e-mails to the login page or would be the URL to the Assessment Office website where there is a hyperlink to the survey's login page.

Project Scope

The scope of the project was mainly transformation of the paper-based survey to web based survey. This includes all necessary features to support the better use of the survey, such as: access validation, data validations, error messages, database tables, etc.

The project would need Hyper Text Mark-up Language (HTML) pages to display the survey questions and as well as the necessary response pages. Besides HTML pages, the project would need ASP pages to properly retrieve information and store survey results in the appropriate table.

During the project, another feature was added to the project, which was not mentioned in the project proposal. Administrator functions were added to the project, so the Assessment Office could easily change the employers and/or graduates data from the

database. The Assessment Office did not request this feature. Even though it was not requested, it was endorsed by the Assessment Office.

The creation of the survey questions was not part of the project. All questions were based on the paper-based survey.

While writing this project report, there was not enough data and time to compare the results of pencil and paper survey and the results of the web based survey. So comparing the survey results is not part of this report.

Here is a simple diagram to help explain how the whole project's architecture works:

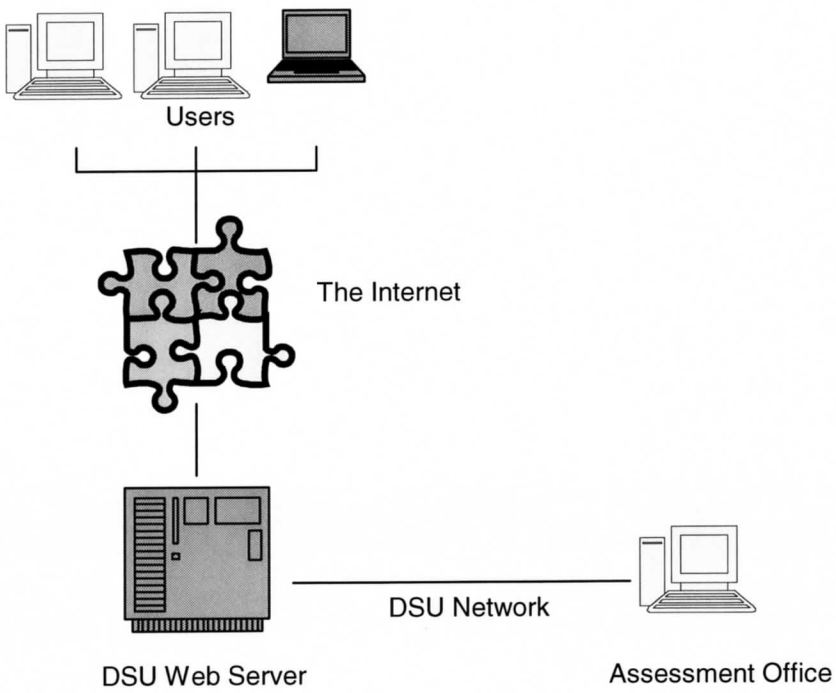


Figure 0.1 System architecture of the project

Chapter 1 Project Components

The project proposal consists of the general direction of the project, but is focused on only providing the web based survey. Carrie Ahern provided me with information about the survey and its background, and helped me to proofread the proposal.

The project planning was based on my previous experience working on Active Server Pages (ASP) and PHP Hypertext Preprocessor (PHP) for course assignments at DSU. The ASP was offered through INFS741 Web Server Management class and the PHP was offered through INFS730 Programming for E-Commerce class. All of the web based features are similar to the method I learned in those classes. The methods used in the project and in the classroom were not exactly the same, so they had to be modified.

1.1 Front End: HTML Form for Survey Questions

Designing and Implementing. All design and implementation tasks were done on the Assessment Office department web space on Dakota State University web server. There were no extra modules or software needed to implement this project.

The first design activity was to design the survey form. I received two sets of paper-based survey questions from Carrie Ahern. The paper-based surveys are available in Appendix A of this report. The questions are displayed in a table so the width of the displayed text can be controlled. All questions are within <table> … </table> HTML tag.

Microsoft FrontPage was used to make the forms. The codes were typed in manually using HTML format, but Microsoft FrontPage was used to change the color scheme.

The table width was set to 640 pixels so if the page is viewed with screen resolution 640 x 480 pixels, the user can view the table without scrolling the screen left to right. The HTML code:

```
<table width="640">
...
</table>
```

The *<table width="640">* starts the HTML table format. The *width* within the *<table>* tag is the properties of the table. In this case, it is 640 pixels. The assumption was that there are still quite few users who use a small format monitor; in this case 640 x 480 pixels screen resolution. The table construction ends with *</table>* closing tag. The screen capture of the survey form can be viewed on Figure 1.1a and Figure 1.1b.

The survey consists of two pages. The second page is only for the education major graduates. The non-education major respondents only needed to answer the first page, but the education graduates needed to answer the second page.

The initial form design was shown to Carrie Ahern. She suggested necessary changes to the form; these changes were made. We changed the set of questions to a newer version for the education employer survey, since the one we used earlier had become obsolete.

As was explained previously, there are two types of user input to answer the survey questions: radio buttons and text areas. A radio button passes a value, either an integer or a string, and a text area passes a string value.

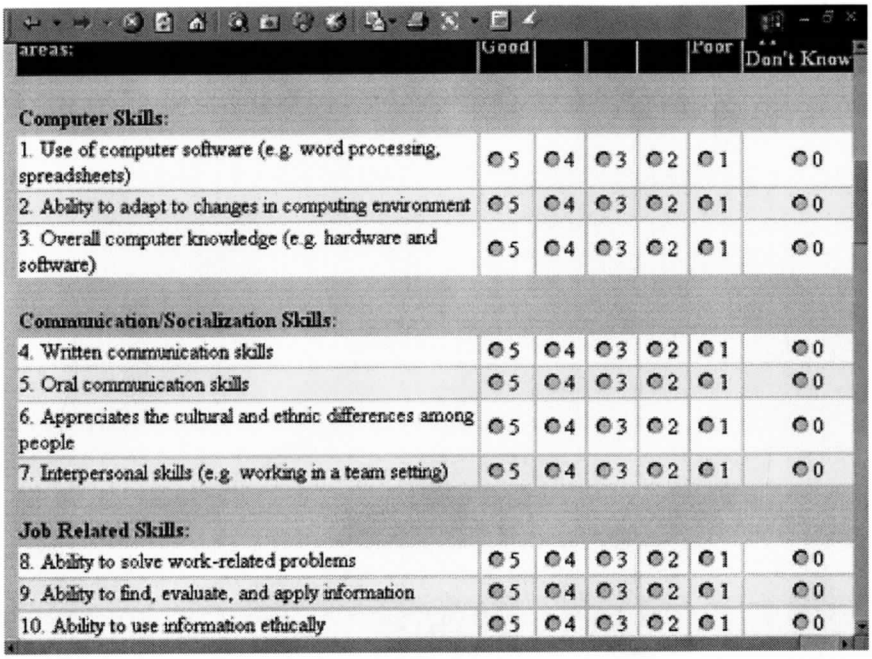
Each survey question needs one field in the database table. The HTML code for the radio button questions is listed below:

```
<tr>
  <td>
    ... the question ...
  </td>
  <td>
    <input type="radio" name="name" value="value1" />option 1
  </td>
  ...
  <td>
    <input type="radio" name="name" value="value6" />option 6
  </td>
</tr>
```

HTML table row starts with *<tr>* tag and ends with *</tr>*. Everything between these two tags will be the information in a row of the table. Table data of a table row starts with a *<td>* tag and ends with *</td>*. A pair of *<td>* and *</td>* tags will create a column in the same row of the table. Options were provided by radio buttons. In HTML radio buttons are constructed with *<input>* tag. Type of input can be text, check box, radio button, password, etc. Since the survey used radio buttons, the input type is radio.

The radio buttons were used for the multiple option questions. We do not need to separate the options with table columns, so we separate them with line breaks *
*:

```
<tr>
  <td>
    ... the question ...
    <br />
    <input type="radio" name="question" value="value1" />option 1
    <br />
    ...
    <input type="radio" name="question" value="value5" />option 5
  </td>
</tr>
```



areas:	Good				Poor	Don't Know
Computer Skills:						
1. Use of computer software (e.g. word processing, spreadsheets)	5	4	3	2	1	0
2. Ability to adapt to changes in computing environment	5	4	3	2	1	0
3. Overall computer knowledge (e.g. hardware and software)	5	4	3	2	1	0
Communication/Socialization Skills:						
4. Written communication skills	5	4	3	2	1	0
5. Oral communication skills	5	4	3	2	1	0
6. Appreciates the cultural and ethnic differences among people	5	4	3	2	1	0
7. Interpersonal skills (e.g. working in a team setting)	5	4	3	2	1	0
Job Related Skills:						
8. Ability to solve work-related problems	5	4	3	2	1	0
9. Ability to find, evaluate, and apply information	5	4	3	2	1	0
10. Ability to use information ethically	5	4	3	2	1	0

Figure 1.1a Screen capture of the survey form using Internet Explorer.

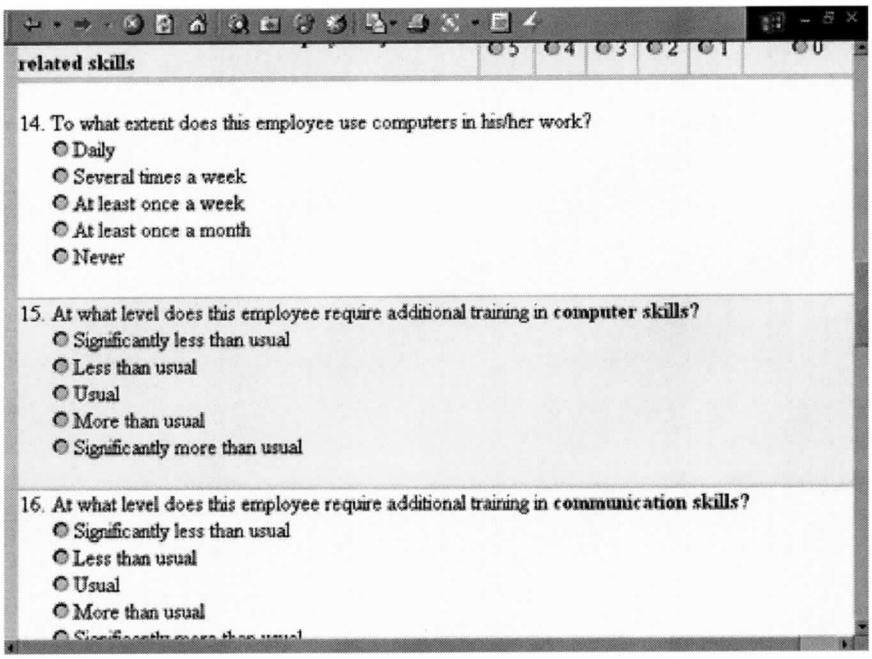


Figure 1.1b Another capture of the survey form.

Here are some examples to explain of the previous codes:

The first three sections of the survey form were using options separated with table columns (shown in *Figure 1.1a*). The following section was using the entire table rows with line break to separate the options (*Figure 1.1b*).

The code to separate the options with table columns looks like this:

```
<html>
<head>
<title>Options separated with table columns</title>
</head>

<body>
<table border="1" width="640">
 <tr>
  <td>
   9. Ability to find, to evaluate, and to apply information.
  </td>
<td>
   <input type="radio" name="q9" value="5" />5
  </td>
  <td>
  <input type="radio" name="q9" value="4" />4
  </td>
  <td>
  <input type="radio" name="q9" value="3" />3
  </td>
  <td>
  <input type="radio" name="q9" value="2" />2
  </td>
  <td>
  <input type="radio" name="q9" value="1" />1
  </td>
  <td>
  <input type="radio" name="q9" value="0" />0
  </td>
 </tr>
</table>
</body>

</html>
```

Every column in the table is constructed with a pair of *<td>* … *</td>* tags. The property of the table is the border and the width. The thickness of the table border is set to 1. To make the table border invisible, the border property is set to 0.

The result of the code in Internet Explorer is shown in *Figure 1.2* below.

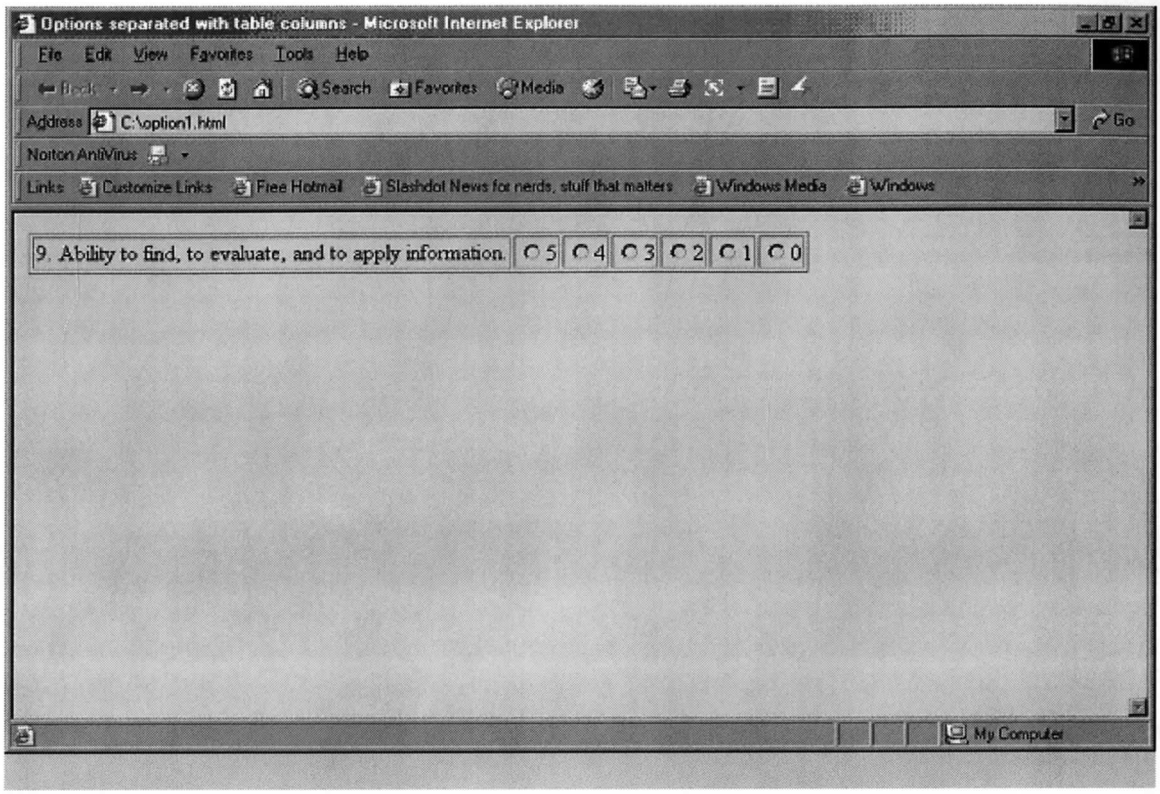


Figure 1.2 Options separated with table columns.

In the previous example, the options are named *q9*. Options for a question should have the same name so that it will have only one answer to be passed, and another page can retrieve the value later on.

The code to separate the options with line breaks looks like this:

```
<html>
<head>
<title>Options separated with line breaks</title>
</head>

<body>
<table border="1" width="640">
 <tr>
  <td>
   18. At what level does this employee require additional training in
       <b>interpersonal skills</b>?<br />

   <input type="radio" name="q18" value="a" />Significantly less than
    usual<br />
   <input type="radio" name="q18" value="b" />Less than usual<br />
   <input type="radio" name="q18" value="c" />Usual<br />
   <input type="radio" name="q18" value="d" />More than usual<br />
   <input type="radio" name="q18" value="e" />Significantly more than
    usual<br />
  </td>
 </tr>
</table>
</body>

</html>
```

With Internet Explorer the code will look like this following figure (*Figure 1.3*).

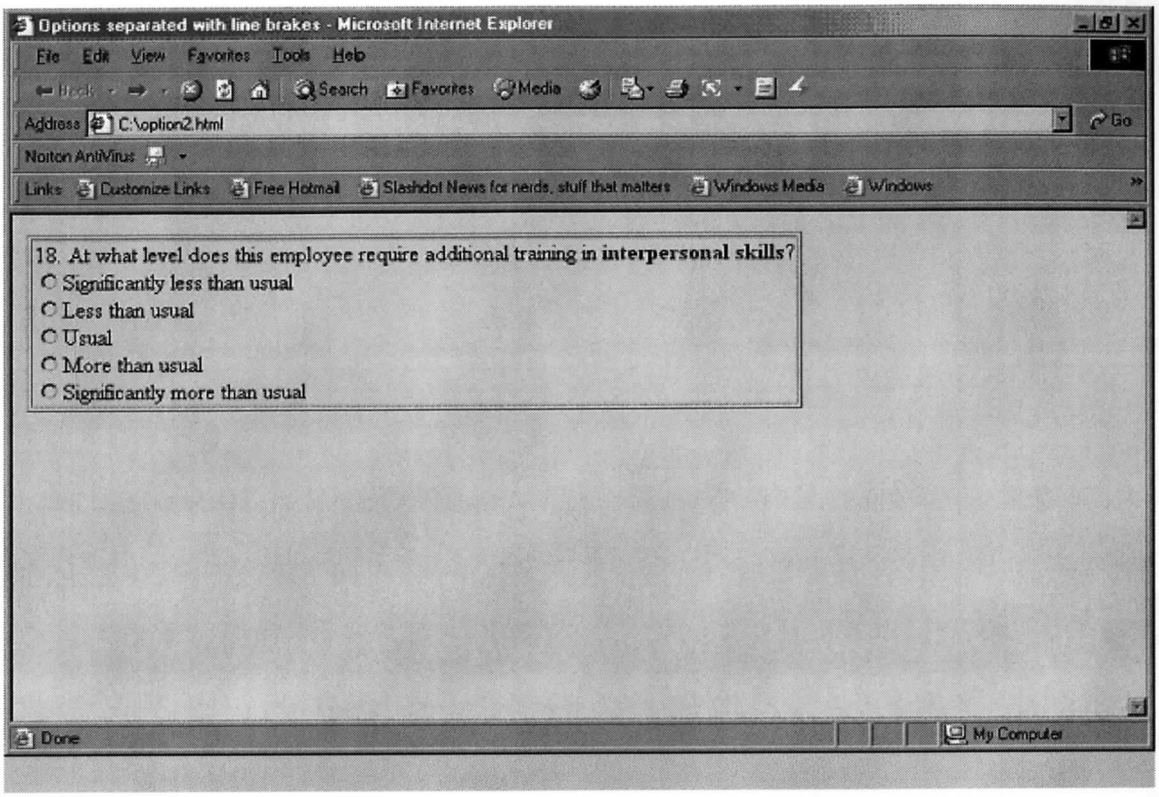


Figure 1.3 Options separated with line breaks.

For the text area input, the code is the following:

```
<tr>
  <td>
    ... question ... <br />
  <textarea name="text1" size="20" onkeypress= "LimitText(this,255)" />
  </td>
</tr>
```

The end part of the *textarea* tag is a JavaScript that calls for a function to limit the number of characters entered into the text area. This is the sample of the JavaScript code to limit the text input:

```
function LimitText(fieldObj,maxChars)
{
var result = true;
if (fieldObj.value.length >= maxChars)
  result = false;
if (window.event)
  window.event.returnValue = result;
return result;
}
```

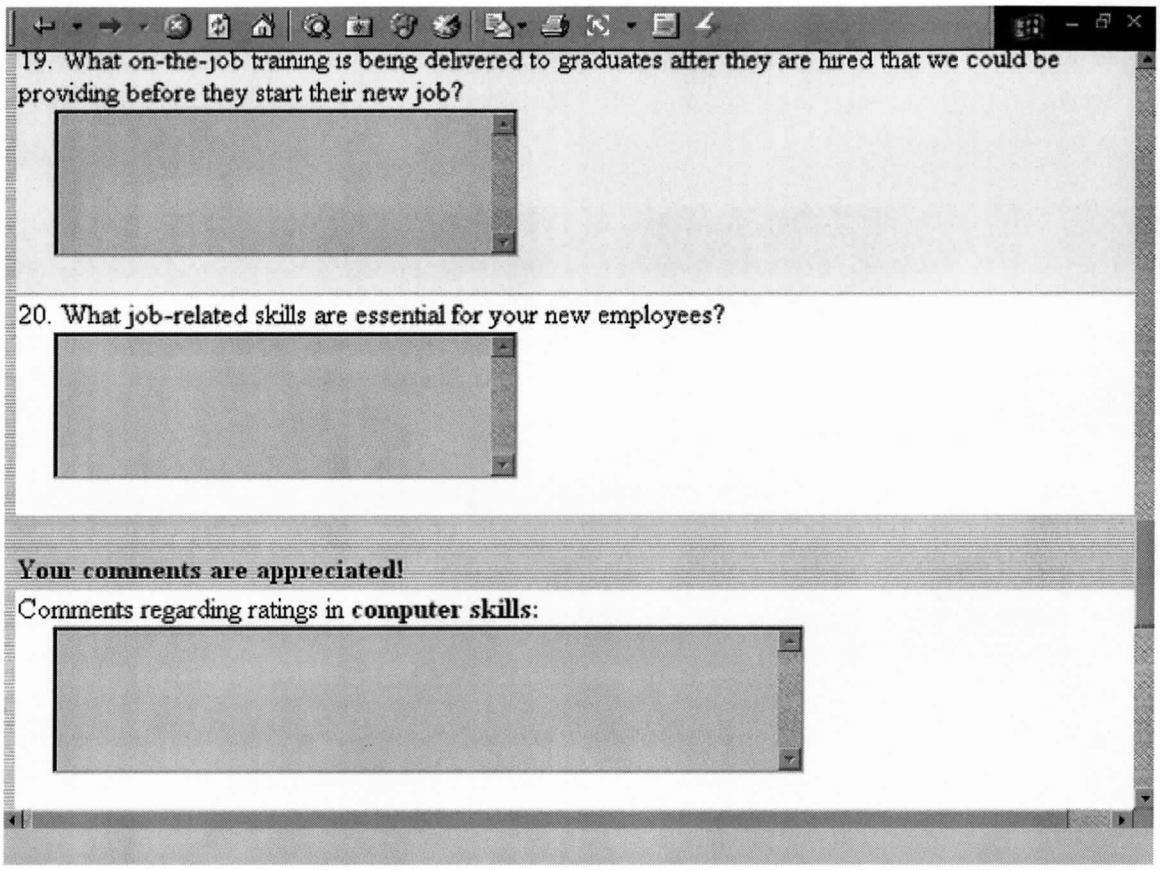


Figure 1.4 Screen capture of text area within the survey form using Internet Explorer.

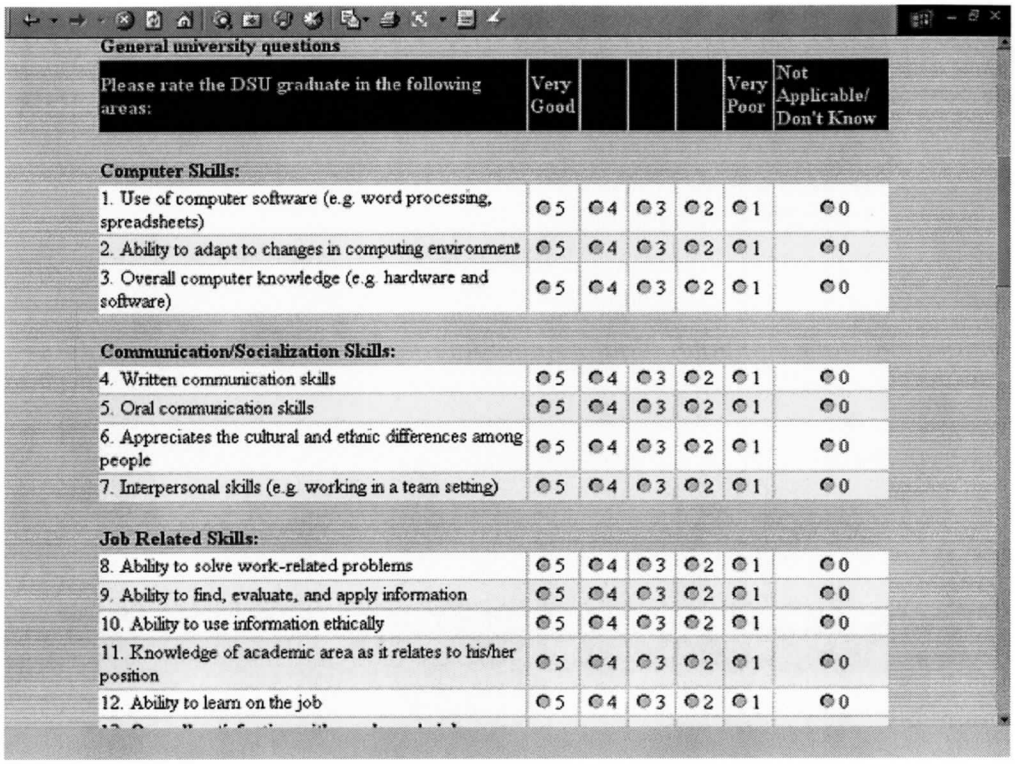
With the JavaScript, the maximum number of characters for the text area is set. The text area is shown on *Figure 1.4*. In this case I chose 255; corresponding to the maximum field size of the database for the related question. The number of character validation is done while the user types his or her comments/suggestion in the text area.

Because of an Assessment Office requirement, the radio button questions cannot be null. The user has to answer the radio button questions. In order to validate that, another JavaScript code is used:

```
function radio_button_checker()
{
// set var radio_choice to false
var radio_choice = false;
// Loop from zero to the one minus the number of radio button selections
for (counter = 0;  counter < survey_form.id1.length; counter++)
{
// If a radio button has been selected it will return true
// (If not it will return false)
if (survey_form.id1[counter].checked)
radio_choice = true;
}
if (!radio_choice)
{
// If there were no selections made display an alert box
alert("Question 1: No answer\n");
return (false);
}
}
```

These JavaScript references are from the Internet. The code was modified so it can be used for the survey form. This block of code is implemented for every radio button question. The id in *survey_form.id* is changed to the corresponding survey question id. The example above is for question number one (id1). The user will get an error message for the first unanswered radio button question after he or she clicks on the submit button.

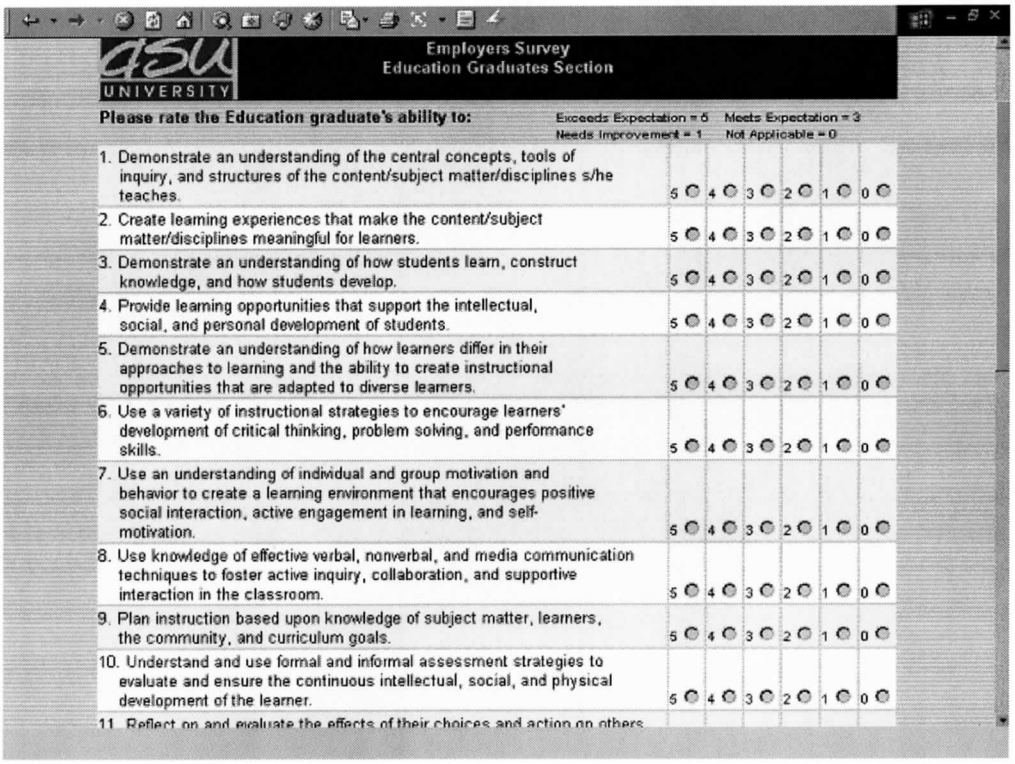
The same error checking method is used for the second page of the survey. The only difference is the number of survey questions. The second page of the survey is almost identical to the first one. The difference can be seen in *Figure 1.5* and *Figure 1.6*.



General university questions

Please rate the DSU graduate in the following areas:	Very Good				Very Poor	Not Applicable/ Don't Know
Computer Skills:						
1. Use of computer software (e.g. word processing, spreadsheets)	○5	○4	○3	○2	○1	○0
2. Ability to adapt to changes in computing environment	○5	○4	○3	○2	○1	○0
3. Overall computer knowledge (e.g. hardware and software)	○5	○4	○3	○2	○1	○0
Communication/Socialization Skills:						
4. Written communication skills	○5	○4	○3	○2	○1	○0
5. Oral communication skills	○5	○4	○3	○2	○1	○0
6. Appreciates the cultural and ethnic differences among people	○5	○4	○3	○2	○1	○0
7. Interpersonal skills (e.g. working in a team setting)	○5	○4	○3	○2	○1	○0
Job Related Skills:						
8. Ability to solve work-related problems	○5	○4	○3	○2	○1	○0
9. Ability to find, evaluate, and apply information	○5	○4	○3	○2	○1	○0
10. Ability to use information ethically	○5	○4	○3	○2	○1	○0
11. Knowledge of academic area as it relates to his/her position	○5	○4	○3	○2	○1	○0
12. Ability to learn on the job	○5	○4	○3	○2	○1	○0

Figure 1.5 Survey form: first page.



DSU UNIVERSITY

Employers Survey
Education Graduates Section

Please rate the Education graduate's ability to: Exceeds Expectation = 5 Meets Expectation = 3 Needs Improvement = 1 Not Applicable = 0

1. Demonstrate an understanding of the central concepts, tools of inquiry, and structures of the content/subject matter/disciplines s/he teaches.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
2. Create learning experiences that make the content/subject matter/disciplines meaningful for learners.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
3. Demonstrate an understanding of how students learn, construct knowledge, and how students develop.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
4. Provide learning opportunities that support the intellectual, social, and personal development of students.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
5. Demonstrate an understanding of how learners differ in their approaches to learning and the ability to create instructional opportunities that are adapted to diverse learners.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
6. Use a variety of instructional strategies to encourage learners' development of critical thinking, problem solving, and performance skills.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
7. Use an understanding of individual and group motivation and behavior to create a learning environment that encourages positive social interaction, active engagement in learning, and self-motivation.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
8. Use knowledge of effective verbal, nonverbal, and media communication techniques to foster active inquiry, collaboration, and supportive interaction in the classroom.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
9. Plan instruction based upon knowledge of subject matter, learners, the community, and curriculum goals.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
10. Understand and use formal and informal assessment strategies to evaluate and ensure the continuous intellectual, social, and physical development of the learner.	5 ○	4 ○	3 ○	2 ○	1 ○	0 ○	
11. Reflect on and evaluate the effects of their choices and action on others							

Figure 1.6 Survey form: second page.

1.2 Back End: Database for Employer Survey

After all the forms were approved, the database design was next. Mr. Tom Farrell, Dr. Zehai Zhou, Ph.D., and Dr. Ronghua Shan helped me with the design on the database. I needed advice for the uniqueness of the survey requirement.

Designing and Implementing. There are two sets of survey questions: One is for education major graduates and the other one is for non-education major graduates.

The data I received from the Assessment Office is in a flat file spread sheet format. It was a Microsoft Excel table that contained the graduate's program. It also contained the following data fields:

1. last name
2. first name
3. name of the company
4. job title
5. last name and first name of contact person
6. company's address (street address, city, state, and zip).

There are four tables in the survey database: company, employee, education survey, and survey. Company table holds information about the company: company name, company address, company id, etc. Employee table is similar to company tables. It contains information about the employee and has company id as its foreign key. Education survey and survey tables both have employee id and company id as the composite key. Both of these tables store the survey results.

PROGRAM	Last Name	First Name	Employer	Title
BSED.ELSPE			Anderson School	2nd & 3rd Teacher
BS.ENGLS			Argus Leader	
BSED.HPE			Armour High School	High School PE
BS.INFO			Associates Commerce Solutions	Dir of I.T. Group
BS.FWM			Avera McKennan Center for Health & Fitness	
AS.RESP			Avera McKennan Hospital	Respiratory Care
AS.RESP			Avera McKennan Hospital	Respiratory Care
AS.RESP			Avera McKennan Hospital	Respiratory Care
AS.RESP			Avera McKennan Hospital	Respiratory Care
BBA.ACCT			Bank First	Data Warehouser
BS.ENGLS			BankFirst	Technical Writer
BS.INFO			BankFirst	Programmer
BSED.ELSPE			Brandon Valley School District	8th Grade Special Ed
BBA.CINFO			Broin & Associates	Programmer
BSED.ELED			Child Center LTD	Pre-School Teacher
BSED.ELED			Children's Care Hospital	Teaching (Mentally I.Child)
BSED.ELSPE			Children's Care Hospital & School	Teacher
BS.BIOLS			Citibank (South Dakota) NA	
AS.APPL			Citibank (South Dakota) NA	Customer Service
AS.APPL			Citibank (South Dakota) NA	Programmer Analyst
BS.INFO			Citibank (South Dakota) NA	Program Analyst
BBA.MKTG			Citibank (South Dakota) NA	Retention Specialist

Figure 1.7 Screen capture of the partial survey data source spreadsheet.

Microsoft Access was used for the database because the Assessment Office staff was already familiar with this database product. Introducing a new database such as MySQL or Microsoft SQL would have required more time to familiarize the staff with the new software. Also, by using Microsoft Access there was no need to install any new module, since the Microsoft Access database connectivity is already supported by DSU's web server.

It was necessary to build two tables to store information about the companies and the employees: company and employee tables. The two tables were populated based on the data from the spreadsheet as shown partially on *Figure 1.7*. The other two tables: education survey and survey tables were built with required fields but left unpopulated. These two tables were left empty because they would store survey results.

The company table and the employee table are shown in *Figure 1.8* and *Figure 1.9*. Those figures also display the stored data within the tables.

COMP_ID	COMP_NAME	COMP_LONGNAME	COMP_ADDR	COMP_CITY	COMP_STATE	COMP_ZIP
		Administrator Account	820 N. Washington Ave	Madison	SD	57042
		Avera McKennan Center for Health & Fitness	800 E. 21st St.	Sioux Falls	SD	57117
		Anderson School	10040 Cottonwood	Bozeman	MT	59718
		Argus Leader	200 S. Minnesota	Sioux Falls	SD	57117
		Sibley East Schools	202 3rd Ave. NW	Arlington	MN	55307
		Armour High School	P.O. Box 640	Armour	SD	57313
		Associates Commerce Solutions	811 E. 10th St.	Sioux Falls	SD	57103
		Avera McKennan Hospital	800 E. 21st St.	Sioux Falls	SD	57117
		Bank First	2600 W. 49th St.	Sioux Falls	SD	57105-6575
		Bank First	1951 N. Alma School	Chandler	AZ	85224-2340
		Brandon Valley School District	301 S. Splitrock Blvd	Brandon	SD	57005
		Child Center Ltd	2420 W. 8th St.	Sioux Falls	SD	57104
		Children's Care Hospital	2501 W. 26th St.	Sioux Falls	SD	57105
		Citibank	701 E. 60th St. N.	Sioux Falls	SD	57117
		Comfort Windows & Doors	2333 Eastbrook Dr.	Brookings	SD	57006
		Community Memorial Hospital	8th and Jackson St.	Burke	SD	57523
		Coolidge Unified School	550 N Arizona Blvd	Coolidge	AZ	85228
		Credit Soup, Inc	106 N. Egan Ave	Madison	SD	57042
		Dakota Beverage Company	606 S. Wayland Ave.	Sioux Falls	SD	57103
		Dakota Ethanol Plant	P.O. Box 100	Wentworth	SD	57075
		Dakota Land Equipment	P.O. Box 10	Madison	SD	57042
		Dakota State University	820 N. Washington Ave.	Madison	SD	57042
		Daktronics	331 32nd Ave.	Brookings	SD	57006
		Delaware N. Peoplesoft Core	Fountain Plaza 10th Floor South Tower	Buffalo	NY	14240
		Dell Rapids School District	1216 N Garfield	Dell Rapids	SD	57022
		Department of Corrections	3200 East Hwy 34 c/o 500 E Capitol Ave	Pierre	SD	57501
		Down Town Chiropractic Center	219 E. 12th St.	Sioux Falls	SD	57104
		DSU Business Office	820 N. Washington Ave	Madison	SD	57042
		DSU Non-Profit Management Inst.	2205 Career Ave.	Sioux Falls	SD	57107
		Dyersburg State Community College	1510 Lake Road	Nashville	TN	38024
		Edmund's Central	105 NE 1st Ave.	Roscoe	SD	57471

Figure 1.8 Company table as developed in Microsoft Access.

The primary key in company table is company id field.

EMP_ID	EMP_LNAME	EMP_FNAME	EDMAJOR	EMP_TITLE	COMP_ID
			☐	Programmer	
			☐	Internet Sales	
			☐	Programmer/Design Dept.	
			☐	Program Associate	
			☑	4th Grade Teacher	
			☐	Programmer	
			☐	Info Tech Analyst	
			☐	Information Systems Tech	
			☐	Assoc Programmer	
			☑	4th Grade Teacher	
			☐	Technical Writer	
			☑	Teaching (Mentally I.Child)	
			☐	Account Analyst	
			☑	2nd & 3rd Teacher	
			☐	Accounting/Office	
			☐	Health Info	
			☑	6th Grade Teacher	
			☐	Accounts Analyst	
			☐	Sofware Developer	
			☐	Coder	

Figure 1.9 Employee table as developed in Microsoft Access.

COMP_ID field in the company table is a foreign key in the employee table. The primary key in employee table is EMP_ID field.

Storing all of the survey results on a single table would not be an efficient design. In a single table approach, the table should be designed to have more fields for the

education major survey results, because the survey has more questions so it would need more table fields to store the results. With this approach for each non-education major survey results, there will be a series of empty fields because the fields are designed only to store answers for education major graduate questions. The solution for this was to make a table to store the education major survey results and another one for the non-education major survey results.

The education survey and the survey table have two primary keys: EMP_ID and COMP_ID. These tables contain all necessary fields to store user answer values for the survey questions. The education survey table holds information about the education major graduates survey and the survey table holds information about the non-education major graduates survey. Both the education survey and the survey table log the IP address of the computer used to fill the survey and when the survey is submitted.

Both education survey and survey tables contain the following fields:

1. employee id
2. company id
3. employee title
4. survey results
5. surveyor position
6. surveyor first name and last name
7. surveyor e-mail address
8. surveyor phone number
9. timestamp
10. ip address

Employee title and surveyor information fields are optional. They can be null.

EMP_ID	COMP_ID	EMP_TITLE	Q1	Q2	Q3	Q4	Q5
		Special Education Teache	4	5	4	4	4
		4th grade	4	4	4	4	5
			4	4	4	4	4
			5	5	5	5	5
		4-8 teacher	4	4	3	3	3
		Middle School Teacher	4	3	3	4	4

Figure 1.10 Part of education survey table after being populated with records.

RES2	RES3	COMP_POS	COMP_FNAME	COMP_LNAME	COMP_EMAIL	COMP_PHONE	TIME_SUBMIT	IP_ADDR
See above.	No suggestions	Supervisor						
Areas of assess	To teach a sem	Past Supervisor						
adequately prep		Supervisor						
adequately prep		Supervisor						
Professional co	To teach the AF	Past Supervisor						
		Supervisor						

Figure 1.11 Another part of education survey table after being populated with data.

Figure 1.10 shows the composite keys: EMP_ID and COMP_ID. Information in any row of this table is identified with these two fields. Figure 1.11 is showing the supplemental information of the survey: the timestamps and the remote machine ip addresses.

1.3 Database Normalization

According to McFadden et al. (1999), normalization is "the process of decomposing relations with anomalies to produce smaller well-structured relations. Normalization is primarily a tool to validate and improve a logical design, so that it satisfies certain constraints that avoid unnecessary duplication of data." (p. 233).

There are several steps in normalization:

First Normal Form. "A relation is in first normal form (1NF) if it contains no multi-valued attributes. Recall that the first property of a relation is that the value at the

intersection of each row and column must be atomic. Thus a table that contains multi-valued attributes or repeating groups is not a relation." McFadden et al. (1999, p. 237).

From *Figure 1.7*, the possibility of a multi-valued attribute is under the *program* field. A graduate can have more than one program field, so the attribute can be multi-valued. To solve that, the database used only a *yes/no* or *Boolean* field to determine whether a graduate is an education or non-education major.

The employee table contains the information about the graduates and consists of these data fields:

1. employee id (primary key)
2. employee last name
3. employee first name
4. education major (yes/no field)
5. company id

The company table contains these data fields:

1. company id (primary key)
2. company name
3. company long name
4. company street address
5. city
6. state
7. zip

Second Normal Form. McFadden et al. (1999):

> A relation that is in first normal form will be in second normal form (2NF) if any one of the following conditions applies:
>
> The primary key consists of only one attribute.
>
> No nonkey attributes exist in the relation (thus all of the attributes in the relation are components of the primary key).
>
> Every nonkey attribute is functionally dependent on the full set of primary key attributes. (p. 237)

Where,

> "A partial functional dependency is a functional dependency in which one or more nonkey attributes are functionally dependent on part of the primary key."(p. 237).

Employee and company tables satisfy the 2NF conditions. The tables each have only one primary key and every other attribute is dependent on the primary key. For the survey and education survey tables, there is one composite key for each table.

The survey table consists of these data fields:

1. employee id (primary key)
2. company id (primary key)
3. employee title
4. survey results fields
5. surveyor position
6. surveyor first name

7. surveyor last name
8. surveyor email address
9. surveyor phone
10. submit time
11. ip address

The nonkey attributes are dependent on the composite keys (employee id and company id).

The education survey table is similar to the survey table. The difference between these two tables is the number of the survey result fields. The education survey table holds more survey result fields. The education survey has more questions than the non-education survey. So the number of fields in the education survey table is more than the number of fields in the survey table.

Third Normal Form. McFadden et al. (1999): "A relation is in third normal form (3NF) if it is in second normal form and no transitive dependencies exist. A transitive dependency in a relation is a functional dependency between two (or more) nonkey attributes." (p. 238)

There are no nonkey attributes in the tables that have transitive dependencies. In other words, all of the nonkey attributes are functionally dependent on either the primary key or the composite keys. All nonkey fields in the company table are dependent on company id fields and all nonkey fields in the employee table are dependent on employee id fields. For the education survey and the survey tables, all nonkey fields are dependent on two set of key fields: employee id and company id fields.

The database relationships between the tables are shown in Figure 1.12.

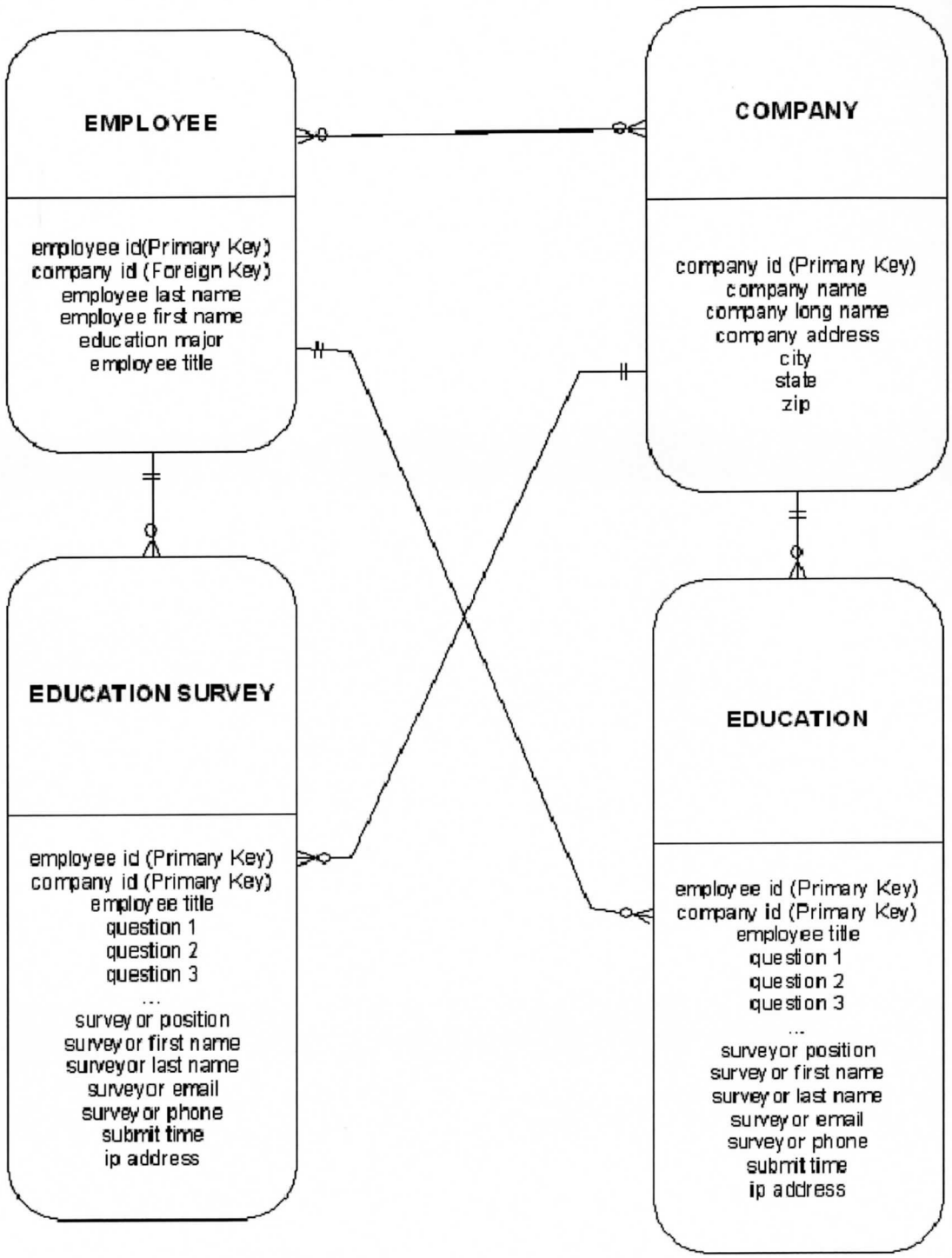


Figure 1.12 Entity Relationship Diagram with attributes

From *Figure 1.12*, one record in the employee table can have zero or many relationships to the company table. This is so one graduate can work at more than one company. A company can have more than one employee.

For the education survey and the survey tables, one record from each table has one and only one relationship to either a record in the employee or the company table. But the employee or the company table can have zero to many relationships to either the education survey or the survey table.

1.4 Metadata

According to McFadden et al. (1999), "Metadata are data that describe the properties or characteristics of other data." In addition to that, "... but do not include that data." (p. 6). Examples of metadata are data type, field length, minimum and maximum values, and descriptions of the data.

Here are the metadata of the tables for this project:

Data Item			Value			
Name	**Type**	**Length**	**Min**	**Max**	**Description**	**Required**
Employee ID	Integer	n/a	1	~	Employee identification number	Y
Employee Last Name	Text	255			Employee last name	N
Employee First Name	Text	255			Employee first name	N
Education Major	Boolean				Education major status	N
Employee title	Text	255			Employee position title	N
Company ID	Text	5			Company identification string	N

Table 1.1 Employee table metadata.

Data Item			Value			
Name	**Type**	**Length**	**Min**	**Max**	**Description**	**Required**
Employe ID	Integer	n/a	1	~	Employee identification number	Y
Company ID	Text	5			Company identification string	Y
Employee Title	Text	40			Employee position title	N
Question1	Integer	1	0	5	Question 1 answer	Y
Question2	Integer	1	0	5	Question 2 answer	Y
Question3	Integer	1	0	5	Question 3 answer	Y
Question4	Integer	1	0	5	Question 4 answer	Y
Question5	Integer	1	0	5	Question 5 answer	Y
Question6	Integer	1	0	5	Question 6 answer	Y
Question7	Integer	1	0	5	Question 7 answer	Y
Question8	Integer	1	0	5	Question 8 answer	Y
Question9	Integer	1	0	5	Question 9 answer	Y
Question10	Integer	1	0	5	Question 10 answer	Y
Question11	Integer	1	0	5	Question 11 answer	Y
Question12	Integer	1	0	5	Question 12 answer	Y
Question13	Integer	1	0	5	Question 13 answer	Y
Question14	Text	50			Question 14 answer	N
Question15	Text	50			Question 15 answer	N
Question16	Text	50			Question 16 answer	N
Question17	Text	50			Question 17 answer	N
Question18	Text	50			Question 18 answer	N
Question19	Text	255			Question 19 answer	N
Question20	Text	255			Question 20 answer	N
Comment1	Text	255			Comment 1	N
Comment2	Text	255			Comment 2	N
Comment3	Text	255			Comment 3	N
Comment4	Text	255			Comment 4	N
Comment5	Text	255			Comment 5	N
Company Position	Text	50			Contact person position	N
Company First Name	Text	20			Contact person first name	N
Company Last Name	Text	20			Contact person last name	N
Company Email	Text	40			Contact person e-mail	N
Company Phone	Text	20			Contact person phone	N
Time Submit	Date/Time				Survey submit time & date	N
IP Address	Text	20			Logged IP address	N

Table 1.2 Non-education survey table metadata.

Data Item			Value			
Name	**Type**	**Length**	**Min**	**Max**	**Description**	**Required**
Company ID	Text	5			Company identification string	Y
Company Name	Text	10			Brief company name	N
Company Long Name	Text	50			Company long name	N
Company Address	Text	50			Company street address	N
Company City	Text	15			City name	N
Company State	Text	2			Abreviated state	N
Company Zip	Text	50			Zip code	N

Table 1.3 Company table metadata

Data Item			Value			
Name	**Type**	**Length**	**Min**	**Max**	**Description**	**Required**
Employe ID	Integer	n/a	1	~	Employee identification number	Y
Company ID	Text	5			Company identification string	Y
Employee Title	Text	40			Employee position title	N
Question1	Integer	1	0	5	Question 1 answer	Y
Question2	Integer	1	0	5	Question 2 answer	Y
Question3	Integer	1	0	5	Question 3 answer	Y
Question4	Integer	1	0	5	Question 4 answer	Y
Question5	Integer	1	0	5	Question 5 answer	Y
Question6	Integer	1	0	5	Question 6 answer	Y
Question7	Integer	1	0	5	Question 7 answer	Y
Question8	Integer	1	0	5	Question 8 answer	Y
Question9	Integer	1	0	5	Question 9 answer	Y
Question10	Integer	1	0	5	Question 10 answer	Y
Question11	Integer	1	0	5	Question 11 answer	Y
Question12	Integer	1	0	5	Question 12 answer	Y
Question13	Integer	1	0	5	Question 13 answer	Y
Question14	Text	50			Question 14 answer	N
Question15	Text	50			Question 15 answer	N
Question16	Text	50			Question 16 answer	N
Question17	Text	50			Question 17 answer	N
Question18	Text	50			Question 18 answer	N
Question19	Text	255			Question 19 answer	N
Question20	Text	255			Question 20 answer	N
Comment1	Text	255			Comment 1	N
Comment2	Text	255			Comment 2	N
Comment3	Text	255			Comment 3	N
Comment4	Text	255			Comment 4	N
Comment5	Text	255			Comment 5	N
Education1	Integer	1	0	5	Education question 1	Y
Education2	Integer	1	0	5	Education question 2	Y
Education3	Integer	1	0	5	Education question 3	Y
Education4	Integer	1	0	5	Education question 4	Y

Education5	Integer	1	0	5	Education question 5	Y
Education6	Integer	1	0	5	Education question 6	Y
Education7	Integer	1	0	5	Education question 7	Y
Education8	Integer	1	0	5	Education question 8	Y
Education9	Integer	1	0	5	Education question 9	Y
Education10	Integer	1	0	5	Education question 10	Y
Education11	Integer	1	0	5	Education question 11	Y
Education12	Integer	1	0	5	Education question 12	Y
Education13	Integer	1	0	5	Education question 13	Y
Education14	Integer	1	0	5	Education question 14	Y
Education15	Integer	1	0	5	Education question 15	Y
Education16	Integer	1	0	5	Education question 16	Y
Respond1	Text	255			Respond 1	N
Respond2	Text	255			Respond 2	N
Respond3	Text	255			Respond 3	N
Company Position	Text	50			Contact person position	N
Company First Name	Text	20			Contact person first name	N
Company Last Name	Text	20			Contact person last name	N
Company Email	Text	40			Contact person e-mail	N
Company Phone	Text	20			Contact person phone	N
Time Submit	Date/Time				Survey submit time & date	N
IP Address	Text	20			Logged IP address	N

Table 1.4 Education survey table metadata.

Chapter 2 Active Server Page Codes

The Active Server Page (ASP) codes were used to make connections between the front end (the HTML form) and the back end of the survey (the database). ASP was chosen because it is a server side script and DSU's web server uses Internet Information Server (IIS) that natively supports ASP. By using server side script, the survey is more secure compared to using client side script (JavaScript, client side Visual Basic script, etc).

All codes were typed in manually through FrontPage. There are a total of 23 ASP files including the survey question forms. During the coding process, I researched how to make the survey form show only the name of graduates that work at a certain company. The names are shown on a list box, so the employer could only choose one graduate at one time.

There was also the problem of security. Anyone could access the survey if he or she knew where to find the location of the survey forms. To solve that problem, ASP session objects are used. The company ID from the login page is saved as a session object on ASP.

According to Buser et al. (2000), "ASP allows the developer to track a user from page to page in an application through the use of a session. A user's session begins when any user without a current session opens any .asp page within an ASP application. The user's session will continue as they navigate from page to page in the site." (p. 312).

Buser et al. (2000) also stated that you can use session objects to "store information that can be accessed by the client throughout the session." (p. 313).

To assign a variable or a string value to a session object you can use the following ASP code:

```
<%
 Session("session_name") = variable name or "string"
%>
```

To set the session timeout, use the following ASP code:

```
<%
Session.Timeout = number_of_minutes
%>
```

The number of minutes should be an integer of how long you want the session objects to persist. After the timeout, all session objects are emptied. To manually clear the session object's values, you can use the following code:

```
<%
Session.Abandon
%>
```

At the beginning of every page, the ASP code checks the session object. The code is like the following:

```
<%
if Session("company_id") = "" then
  response.redirect "timeouterror.asp"
end if
%>
```

In the example, the session object to be checked is company_id. The value "" is actually a null value. By using an if…then statement, the session object is checked and if it contains a null value then the user would get to the error page, timeouterror.asp.

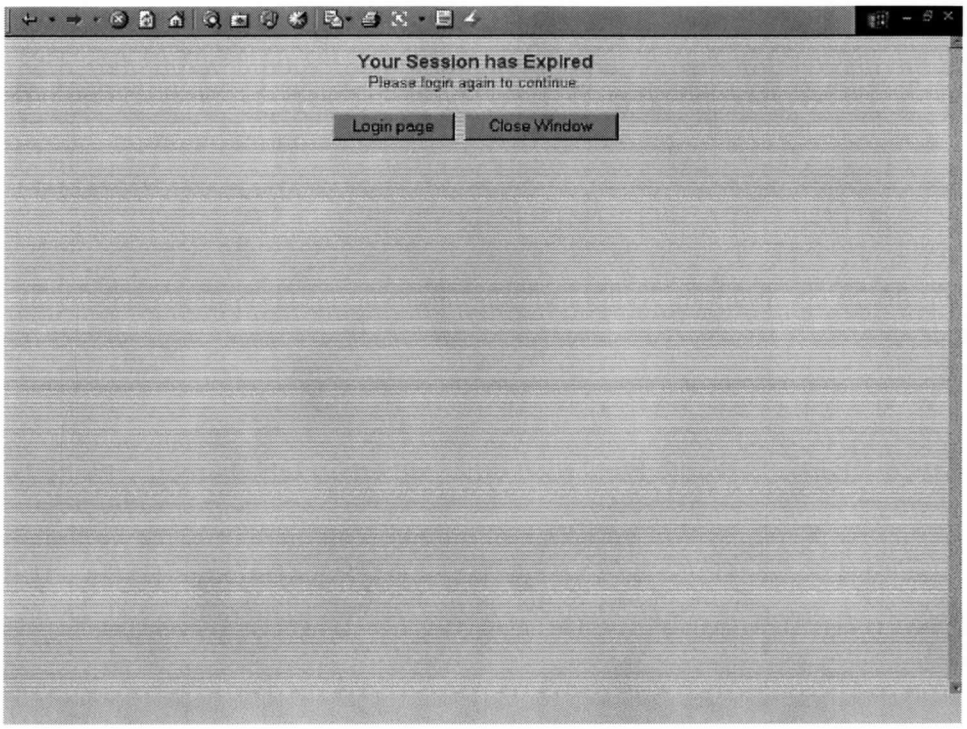


Figure 2.1 Expired session error page.

The session object time out was set to 30 minutes. On every page, the company ID session object will be checked. If it is empty (null), the user will not be able to access the page. This is also true if a user successfully logs in and his or her session times out. Figure 2.1 shows the error page if a user has an expired company id session object. The method has one drawback: the user has to be connected to the web server at all times while he or she fills out the survey.

The cookie solution is not suitable for security reasons and we do not want any survey information stored in a user's computer. Debugging was done after all the main functions: the HTML form and the database were tested and worked without any error. Debugging included checking the page flows, checking the query strings, and simulating the survey entry.

Data types have to be consistent throughout the page. The variables in the ASP had to contain the corresponding data type within the database before the record entry. The same rule was applied in the query string. The data type in the query statement must correspond with the one in the table.

2.1 Retrieve Data from the Database

The first connection the page needed was the login for the survey. The user will need the company id and a password to get in to the survey. This information is stored in the database. User input then is compared with the value stored in the database. The user could only access the survey when the company id and password matched the database

Server side include (SSI) was used to make the database connection because the connection occurred more than once in multiple pages. The HTML code for SSI is:

```
<!-- #include file="connectivity path/filename" -->
```

In the database connectivity file:

```
<%
dbname="database_file_path"
set conn=server.createobject("adodb.connection")
cnpath="DBQ=" & server.mappath(dbname)
conn.open "DRIVER={Microsoft Access Driver (*.mdb)}; " & cnpath
%>
```

This code opens the connection to the database, so the page can retrieve or make queries from the database. The *dbname* is the variable that contains the path to the location of the database file for the connection. The connection object name is *conn* and that sets the type of connection. In this case, it was an ADODB connection. The

connection path (*cnpath*) is the actual location of the database file in the web server. The connection was opened by supplying the connection object (conn) with the driver to access the database and the actual path to the database file.

From the login form:

```
<form method="POST" action="response_file.asp">
Company ID<input type="text" name="compID" size="10"
tabindex="1"><br />
Password: <input type="password" name="password" size="15"
tabindex="2"><br />
   <input type="submit" value="Submit" name="Submit">
   <input type="reset" value="Reset" name="Reset">
</form>
```

The *<form>* and *</form>* tags start and end an HTML form. In the *<form>* tag there is an *action* property to assign where the user will go after he or she clicks on the submit button. There were two inputs for this form: text box for the company id, and the password text box.

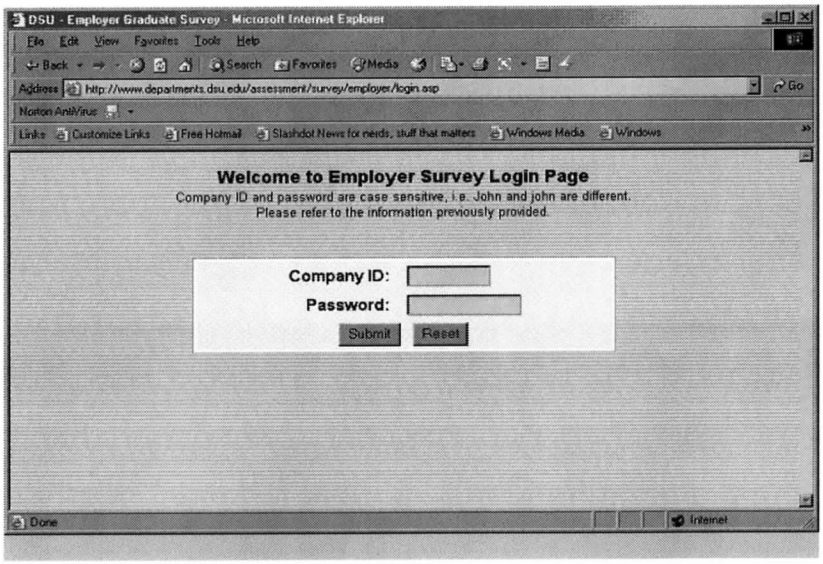


Figure 2.2 Login page.

Every input has its own name, so it can be retrieved later. There are also two command buttons, one to submit the login information and another one to clear all input text boxes.

The response file for the login page gets the company id and password from the user input through the form. This information is then compared with the values in the database.

```
<%
' get company id and password
companyID = request.form("compID")
passwd = request.form("password")
Session("ID") = companyID
%>
```

The code obtains the values from the previous form with *request.form("form_item_name")*. Single quote sign (') starts a line comment in ASP. Comments were used to make the code maintenance and upgrade tasks easier to do. The companyID variable is compared with the retrieved record from the database. If the database does not have any record of companyID, then the page will close the connection and will give the user an error message. Series of *if...elseif...then* statements and redirections are used.

After the company id and user password are checked and validated, a session object for the company id will be created. The company id session object was used as the key information to retrieve any record from the database, such as company's address, graduates who work for the company, etc.

Retrieve Employees. On the first page of the survey there is a drop down list box of names of employees who work at the corresponding company (See *Figure 2.3*). This will depend on the login information. Company id session object is used to clarify the login information.

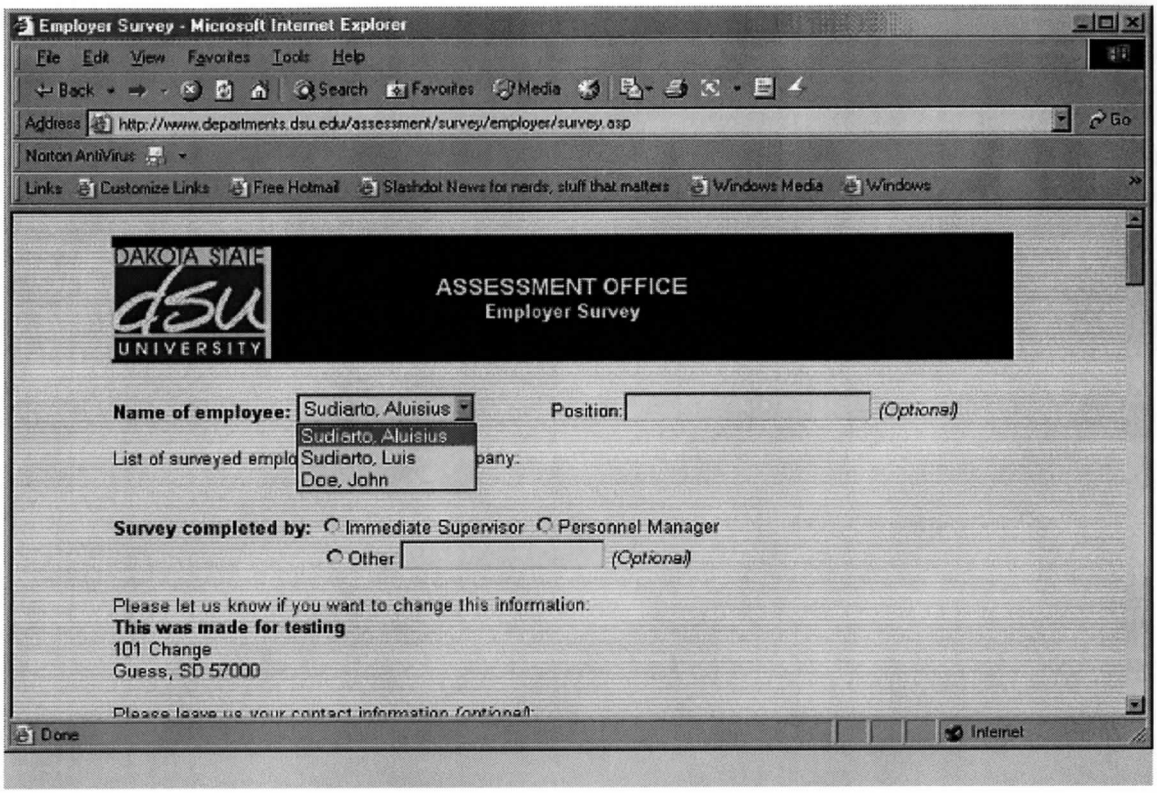


Figure 2.3 Employee and company information on the survey page.

The first step was to query the database for employee's id, last name, and first name of the employee who worked at a certain company, which is stored in a session object. The second step was to make a loop to get all of the records and put them in a list box and to assign each employee id as its value. The code might look like this:

```
<%
' get employee names from employee table
companyID = Session("companyID")
sqlstate="select EMPLOYID, LASTNAME, FIRSTNAME from employee where
employee.COMPANY_ID='" & companyID & "';"
set rs = conn.execute(sqlstate)
Do While NOT rs.EOF
  empID = rs("employid")
  empFName = rs("firstname")
  empLName = rs("lastname")
%>
<OPTION VALUE="<%= emploid %>">
<%= lastname & "," & " " & firstname %></OPTION>
<%
rs.movenext
Loop
%>
```

The ASP variable companyID gets its value from the companyID session object. Then the ASP variable was used to query the employees who work for the particular company id. Queried table name is employee. The queried fields are employed, lastname, and firstname. The query looks for any records in the employee table where the company id field is the value of variable companyID. The *rs* is the name of recordset object to query the database. The queried values are then assigned to variables to be displayed in the drop down text box. The drop down is created by using <option> ... </option> HTML tags. The Do While... Loop was used for multiple employees. The loop creates the entire drop down options from the queried instances, one at a time until the end of the query result.

Retrieve Company Address. Retrieving the company address was the simplest query in this project. Based on the session object that contains the company id, the page queried

the database for the company name, street address, city, state, and zip from the company table. The code might look like this:

```
' get company information
  sqlstate="select * from comp_table where COMPID='" & companyID &_ "'"
set rs=conn.execute(sqlstate)

compName = rs("company_name")
compAddr = rs("address")
compCity = rs("city")
compState = rs("state")
compZIP = rs("zip")
```

The queried field values are assigned to variables. The variables then displayed to the browser with *response.write variable_name* and/or with the short cut <%= *variable_name %>* (*Figure 2.3*).

2.2 Store/Update Data to Database

Before storing the survey data to the database, some of the survey data values must be transformed. Some of the survey questions have a long phrase as an answer. To save coding time and space, a simpler value for the options was used in the HTML form. The value was changed to its original value before it stored in the database. There were some survey questions that have the same options (question 15 through question 18 on the non-education section). It was easier to transform the form values than set the values to the corresponding phrases. For example look at the first page of the survey, question number 14 in the appendices or *Figure 2.4*.

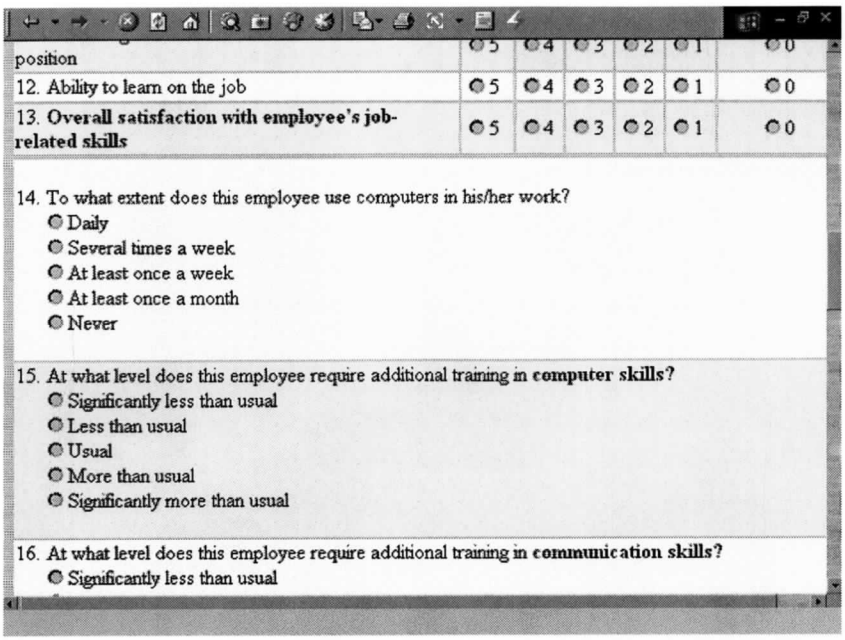


Figure 2.4 Survey questions with transformed option values

```
' get survey question values
id(14) = Request.QueryString("id14")

' data transformation
' question 14
Select Case id(14)
Case "a"
Q(14) = "Daily"
Case "b"
Q(14) = "Several times a week"
Case "c"
Q(14) = "Once a week"
Case "d"
Q(14) = "Once a month"
Case "e"
Q(14) = "Never"
End Select
```

Value a, b, c, d, and e are values from the radio buttons in HTML form.

The education survey is stored in education the survey table and the non-education survey is stored in the survey table. It is required to check whether the surveyed employee is an education major or not before storing the survey into a certain table.

Although there is only one survey, it was necessary to have two different tables for each of the education survey and the non-education survey. The number of survey questions is different between the education survey and the non-education survey. The education survey has an extra set of question besides the set of questions that the non-education survey has.

If a single table was used for both education and non-education graduates, there would be a lot of empty fields in the table, because the extra set of question for the education survey would be left empty for the non-education survey.

The education major information is retrieved from the database. The field is a Boolean field, so it returns *true* or *false* value (see Figure 2.5). An if…then statement is used to store the survey into an appropriate table.

EDU_MAJOR	EMPLOYEE_TITLE
☐	Programmer
☐	Internet Sales
☐	Programmer/Design Dept.
☐	Program Associate
☑	4th Grade Teacher
☐	Programmer
☐	Info Tech Analyst
☐	Information Systems Tech
☐	Assoc Programmer
☑	4th Grade Teacher
☐	Technical Writer
☑	Teaching (Mentally I.Child)
☐	Account Analyst
☑	2nd & 3rd Teacher
☐	Accounting/Office

Figure 2.5 Part of the employee table: EDU_MAJOR field.

```
' set input table according to education major status
empEdMajor = rs("EDU_MAJOR")
if (empEdMajor) then
tableName = "education_survey"
else
tableName = "non_ed_survey"
end if
```

The selected check box on the education major field (EDU_MAJOR) return the value true and the unchecked ones return the value false. Those values are used with the if…then statement to determine into which table the survey results will be stored. All the survey results are stored in either the education survey table or the non-education survey table, in the example the table names are the education_survey and the non_ed _survey respectively.

A *recordset object* with ActiveX Data Objects (ADO) connection was used to store the survey to the database. According to Buser, et al. (2000), "ADO is an interface that allows our ASP pages to talk to OLE-DB." and "ADO is a higher-level model than OLE-DB, which means that it simplifies some of the complexities of programming with OLE-DB. Thus, ADO is much easier to use than OLE-DB." (p. 468). Use this code to make the connection for data entry:

```
' string to supply connection info
strConn = "DRIVER={Microsoft Access Driver (*.mdb)}; " & _
"DBQ=" & Server.Mappath("database_file_path") ' string to supply connection
info
' set recordset and insert record
set oRS = Server.CreateObject("ADODB.Recordset")
oRS.open tableName, strConn, 3, 3
```

The constant after the connection string are the cursor type of the connection and the second constant is the lock type. I will explain these more in the next chapter since they are part of the database connectivity.

It is necessary to check for any attempts to make a duplicate entry. If a graduate is already surveyed, there will be no new values allowed to be stored in the database. On the first survey page, the name of the surveyed graduates is displayed to let the user know every graduate from his/her company that is already surveyed (*Figure 2.6*).

```
' check for attempt of duplicate entry
while not oRS.eof
empIDTable = CStr(oRS("employeid"))
if empIDTable = employeID then
error = error + 1
end if
oRS.movenext
wend
```

This will find any attempt to add a duplicate entry survey result into the database.

Figure 2.6 shows an example of an attempt to create a duplicate survey entry for employee Aluisius Sudiarto. After the user submitted the survey, he or she will get the error page shown in *Figure 2.7*.

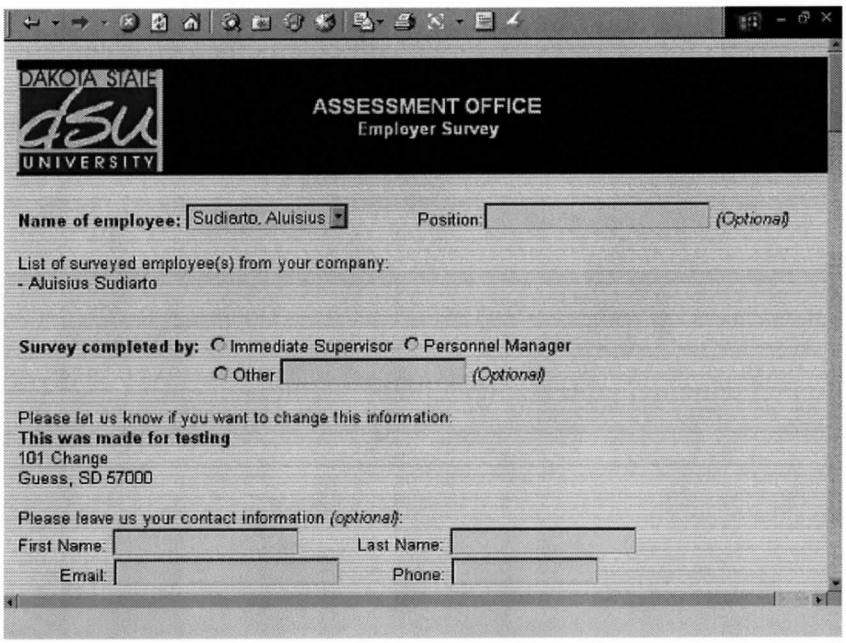


Figure 2.6 Attempt to create a duplicate entry.

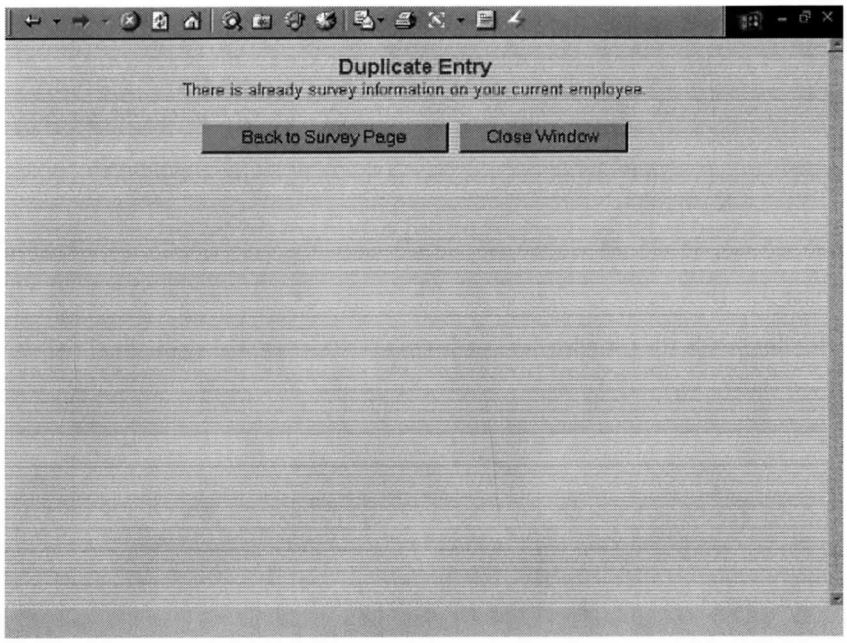


Figure 2.7 Duplicate entry error page.

To store the survey to database, I used this:

```
' table survey entry
oRS.AddNew
oRS("field_name1") = value1
  oRS("filed_name2") = value2
  oRS("field_name3") = value3
  ...
oRS.Update
```

The recordset object is oRS. It was set when creating the database connection. Between oRS.AddNew and oRS.Update are the data entries. The entry values have to correspond with the data type of the fields, string fields require string value, integer fields require integer value, so on and so forth. The table will be updated with oRS.Update. The string value will need to be within double quotes and integer or number values, arrays, and any other data types do not need double quotes.

After storing the first survey form to the database, the user should be redirected to the second page if necessary. It depends on the graduate's major.

```
<%
if tableName = "education_survey" and error = 0 then
Response.Redirect "education_survey.asp"
elseif tableName = "non_ed_survey" and error = 0 then
%>
-- successful submit message—
```

If the graduate is an education major, the survey result is stored partially into the database. Because education major graduates have an extra set of questions, the database for the corresponding graduate will be updated after the user submits the second survey form.

The step is similar to the first survey form. The survey result from the survey form is stored in the education survey table; then the education table with survey result is updated with the second survey form for the corresponding employee id.

A new connection to update the table was required. Since it only needs to update one table, it can be done with a single explicit connection. A connection object was used to do this. I used the same connection string as in the first survey form (p. 34). The update process was done by executing a SQL statement string:

```
' sql statement to update the survey information
sql_statement = "update education_survey "
sql_statement = sql_statement & "set field1 ='" & value1 & "', "
sql_statement = sql_statement & "field2 ='" & value2 & "', "
...
sql_statement = sql_statement & " where ... "
```

To execute the SQL statement:

```
set rs = conn.execute(sql_statement)
```

Checking for duplicate entry was not necessary since it was already being done before submitting the first survey form.

Chapter 3 Database Connectivity and SQL Statements

3.1 Database Connectivity.

ActiveX Data Object (ADO) was used rather than Open DataBase Connectivity (ODBC) because it is more advanced. It is not necessary to make the data link to the database file, if the ADO connection is used.

According to Buser, et al. (2000), "The most obvious way to get some data from a data store is to explicitly create an ADO Connection object in our code, and then use the Connection object's properties when working with the recordset." (p. 513).

The connection object is used to make the database connection:

```
dbname="database_path"
set conn=server.createobject("ADODB.Connection")
cnpath="DBQ=" & server.mappath(dbname)
conn.Open "DRIVER={Microsoft Access Driver (*.mdb)}; " & cnpath

set rs=server.createobject("ADODB.Connection")
set rs.Execute (sql_statement_string)
```

An inexplicit connection object was used for connection to the recordset object directly. The direct method was used to insert records into the database. The code is less confusing by using a direct way, since you do not have to explicitly make the connection to the database.

```
' string to supply connection info
strConn = "DRIVER={Microsoft Access Driver (*.mdb)}; " & _
        "DBQ=" & Server.Mappath("database_path")
        ' string to supply connection info

set oRS = Server.CreateObject("ADODB.Recordset")
oRS.open table_name, strConn, 3, 3
```

To make a number of queries to the same data store, use the explicit connection by using the connection object. To make only a query to the database, such as storing new data, use the direct connection by using a recordset object.

The numbers, right after the connection string in the code above, are cursor type and lock type respectively. If there is a third value, it is the recordset's command type. Command type is optional.

Cursor Type. What is the cursor type?

> "We can think of the recordset's cursor as a type of pointer. Every recordset has exactly one cursor, which (at any given time) points to exactly one of the records in the recordset." Buser, et al. (2000) p. 522.

You can use one of these cursor type constants:

Constant	Value	Description
adOpenDynamic	2	A dynamic cursor with both forward and backward scrolling where additions, deletions, insertions, and updates made by other users are visible
adOpenForwardOnly	0	Default, a forward scrolling only, static cursor where changes made by other users are not visible
adOpenKeyset	1	A keyset cursor allows you to see dynamic changes to a specific group of records but you cannot see new records added by other users
adOpenStatic	3	A static cursor allowing forward and backward scrolling of a fixed, unchangeable set of records
adOpenUnspecified	-1	Cursor type not specified

Table 3.1. **Cursor Type table.**[3]

[3] www.devguru.com

Lock Type. According to Buser, et al. (2000) p. 525 – 526,

The locking type is closely related to whether or not the recordset is updateable. If you are using your recordset to query a data store, then the recordset is a copy of the records that you requested in the data store. That means that there is one copy of the record in your recordset and another copy of the record in the data store. Then, updating a record is a two stage process:

- First, you edit the copy of the record that is contained in your recordset
- Second, you update your changes to the copy of the record that is in the data store

You can choose to lock the data store copy of the record while you are making your changes – and keep other users from touching them.

Constant	Value	Description
adLockBatchOptimistic	4	Multiple users can modify the data and the changes are cached until **BatchUpdate** is called
adLockOptimistic	3	Multiple users can modify the data which is not locked until **Update** is called
adLockPessimistic	2	The provider locks each record before and after you edit, and prevents other users from modifying the data
adLockReadOnly	1	Read-only data
adLockUnspecified	-1	Lock type unknown

Table 3.2. **Lock Type table.**[4]

[4] www.devguru.com

3.2 SQL Statements

All database modifications from the web pages are done with SQL statements within the ASP script. This includes database records retrieval. Use SQL statements to retrieve, to update, and to delete records from the database.

SQL statements are executed through recordset properties.

```
'connect the database
dbname="database_file_path "
set conn=server.createobject("adodb.connection")
cnpath="DBQ=" & server.mappath(dbname)
conn.Open "DRIVER={Microsoft Access Driver (*.mdb)}; " & cnpath

'set the sql statement
sql_statement = "... sql_statement string..."

'execute the sql statement
set rs = conn.execute(sql_statement)
```

First, you need a connection to the database. The example uses *conn* as the connection object to make the connection. Second, create a query string. The *sql_statement* is the variable that holds the query string. And finally, execute the query. The query result will be in the *recordset* object, *rs*.

Retrieve Record. To check whether the company id and password are valid in the login page, you can query the company table for the information.

```
sql_statement = "SELECT company_id, password FROM company WHERE
company_id = '"& <%= companyID %> & "'"
```

This statement is the sql statement string. It was stored in an ASP variable and then executed. The variable companyID is an ASP variable that contains the company id from the HTML login form. Since the ASP variable itself is not part of the SQL statement, it is

located outside of the double quotes (“) signs. ASP assigns the value of the variable companyID that contains the company id of logged in user. The company id is a string, single quotes (') are needed within the SQL statement.

The queried fields are specified in the SQL statement. The example queried the database field: company_id and password. The target table name of the query is company. The ampercent (&) sign was used to concatenate or combine the string.

If there is no such company id, the *recordset* will return *eof* (end of file). If the recordset found the company id in the database, the password from the login form and from the database is compared. According to these rules, series of if…then statement are made to direct the page flow.

Another example of data retrieval from the database is to query the company information:

```
sql_statement = "SELECT * FROM company WHERE company.COMPANY_ID =
'" & <%= companyID %> & "'"
```

Execute this SQL statement and retrieve the information with ASP script:

```
<%
companyName = rs("COMPANY_NAME")
companyAddress = rs("COMPANY_ADDRESS")
...
%>
```

In this example the queried fields are not specified within the SQL statement. Instead there is a wildcard expression (*). It means the query needs all the fields within the targeted table, company. ASP recordset is used to retrieve the specific field value by

specifying the field's name, COMPANY_NAME and COMPANY_ADDRESS. The field values are assigned to ASP variables, companyName and companyAddress.

Update Record. The record was updated using SQL statements (see p. 45).

An update was needed to add education survey results to the already stored non-education survey results in the education survey table. The condition of the update was based on the employee id and the company id. The process will only update the records with a certain employee id and company id fields combination.

```
sql_statement = "UPDATE table_name SET field1 = 'value1', field2 = 'value2'
WHERE EMPLOYEE_ID = '" & employeeID & "' AND COMPANY_ID = ' " &
companyID & "';"
```

The SQL statement to update a table can be a long string. It depends on how many fields are being updated. For this example field1 and field2 were updated with value1 and value2. The statement will update a record if the conditions are met. In this case, the conditions - value of EMPLOYEE_ID and COMPANY_ID fields- have to be the value of ASP variable employeeID and companyID respectively.

Delete Record. The SQL statement:

```
sql_statement = "DELETE * FROM table_name WHERE field1 = value1"
```

This SQL statement will remove a record from the database if the condition is met. The value can be a string. The condition of the SQL statement is field1 contains value1. If the value is a string, the SQL statement will be:

```
sql_statement = "DELETE * FROM employee WHERE EMPLOYEE_ID =
'EMP001'"
```

This SQL statement will delete all fields of a record that has EMPLOYEE_ID field value EMP001 from the employee table.

Since the process is destructive, only the administrator account can delete database records. The key field of the table is used to remove a record. For example: To remove an employee record from the employee table, you will need the employee id.

3.3 Administration

The administration account was made, so that later on it will be easier to modify the database. The administration function is a separate module from the survey. The survey can run just fine without the administration function, but it would take less time to modify the database through the web based administration function.

Previously to make any changes to the database, database maintainer/administrator needed the FrontPage to be connected to the web server and to open the database file. This process depended on how good the connection speed was between the local machine and the web server. In short, the administrator needed to download the Access database file, make the necessary changes, and then upload the file back to the web server. Since the database was growing, maintaining the database would be a time consuming task.

By providing a web based administration function, it would take less time to maintain the database and it could be done from anywhere in the world through an

Internet connection. The web based administration only needs a browser to make changes to the database.

Currently, the administrator account can only add, delete, and edit records within the company and the employee table. It would need more detail requirements from the Assessment Office to be able to query the survey tables if necessary.

Chapter 4 Extension Possibilities and Future Work

Implement Database Driven Survey Form Method. As mentioned previously, all of the survey questions for both pages were typed in HTML form. It would be better if a database was used for the survey question so it will be easier to update the survey questions later. The method would be similar to a database driven web page.

According to mysql.com[5], the database driven web page has some benefits, "By achieving complete separation between your site's design and the content you are looking to present, you can work with each without disturbing the other." Content management in web driven web page allows the content writers to update the page themselves without have to know about HTML codes.

The database would store all of the components of a page, in this case question types, the survey questions, the multiple option answers, the value of the options, etc. To update the page, user could use the provided HTML forms. The survey questions can be updated without touching the displayed HTML page format and without changing the page layout. Because the survey form would be generated from the database, it will not be necessary to change the HTML layout.

Besides the four tables that contained the information about the company, the employee, and the survey result, there would be more tables to store information about

---

[5] http://www.mysql.com/articles/ddws/

the survey itself. These tables would not necessarily related to the other four already existed tables.

Database System Migration. There are some more advance database systems than Microsoft Access. Better yet, some of them are free. It depends if the survey is used in a centralized database where employee and company records are stored in part of one huge DSU database. In choosing which system to be used, we should consider the compatibility of the survey database with the main database system.

For a simple project, like the current online survey, Microsoft Access had satisfied the database system requirement. But it has limitations to a certain degree. According to clearform.com[6], as directly quoted, MS Access has these following disadvantages:

> *Limited to small databases*
>
> Microsoft Access is a low capacity database system. It is designed to efficiently manage relatively small numbers of database records. Although it lacks in high volume capacity, it makes up for this limitation with ease of use features, "Rapid Application Development" technology, and portability.
>
> Microsoft Access 2000 has a 2 gigabyte file size limit for an entire database (MDB file). Although you may be able to append hundreds of thousands of records into MS Access database tables, performance will noticeably began to

---

[6] http://www.clearform.com/microsoft_access.htm

decrease when you reach a volume of approximately 25,000 records. Performance will decrease even further as you add more records and/or additional concurrent users.

The ultimate effect that a large volume of records has on performance is partly influenced by the database design. A Microsoft Access table record can contain anywhere from 1 field to 255 fields. Thus, a 2 field record will require a lot less resources than a 50 field record. The type and form of data being stored in a record also effects performance. Pasting a large JPG or GIF image into a record, could make the size of one image record be equal to 50 or more "text only" records.

The good news is that efficient database design, data management, and archiving techniques can often work around the capacity limitations of Microsoft Access.

Multiple survey results from different years can be stored into the same database file. The same database file can be use throughout the lifetime of the system. The number of the table might not change, but the number of rows within the table definitely will grow in accordance to the number of survey submissions.

Currently, the database file size for the survey is extremely small compared to the Microsoft Access limitation. But in the future, this limitation would prevent storing the new survey results, or additional companies or graduates.

More on the Microsoft Access limitations:

Limited to Low Transaction Rates

Using Microsoft Access for database design, allows the developer to utilize a wealth of input validation, formatting, data display, querying, and reporting tools. As far as user-friendliness, these tools are unparalleled by any other database system currently found on the market. Developers can quickly design database systems that provide maximum ease of use for end-users. Quick development and ease of use are the two primary advantages of choosing Microsoft Access for your database design and use.

However, pre-built user-friendly tools come with a downside. Microsoft Access was designed with the primary goal of providing ease of use for inexperienced and experienced database users/developers. To accomplish this goal, Microsoft had to use a one file system and embed many of its ease of use tools into the actual MDB file, data tables, and form objects. In fact, all of the objects and tables reside within one file format. This type of structure results in Microsoft Access performing sluggishly in high production environments, which require large volumes of data input per hour. Microsoft Access is definitely not a good choice for high production data entry environments.

With the relevant growth of the survey, the rate of survey submission will also increase. It is better to consider the limitation on how many survey submissions that the database can handle within the certain amount of time. Currently there is no problem with the rate of the submission. The number of company and graduates were considered to be

a small amount and still within the database limit. But for the future use, it is wise to consider this limitation.

Limited to Low Concurrent Database Usage

Microsoft Access runs best as a single-user system or a small multi-user system. Although, there are claims that 50 or more users can use Microsoft Access concurrently with no problems, this is not always recommended. As mentioned previously, the structural design of Microsoft Access will limit performance in high production environments. This also applies to high levels of concurrent usage, especially concurrent data entry operations. If concurrent usage does not entirely involve data entry operations, then higher concurrent usage may be possible without performance issues. For example, 10 users running reports and read-only queries may not conflict with an additional 5 -10 users inputting data at the same time. Testing the various scenarios in your computing environment is the best way to determine what is optimal. Keep in mind that your hardware and operating system capability will also be a factor.

The future progress of this project might include built-in analytical and statistical tools. These tools should run concurrently while the database receives the survey submissions, in order to obtain the analytical and statistical reports instantly. These report features need complex queries or operations within the database. Such activities will push the database to its limit to handle so many concurrent requests from the user.

Changing the database system to a better system would depend on the situation. So far, the survey has not had any problems with either the web traffic or the database limitations. In the future, the database limitations should be put into our consideration. With the database growth of usage and complexity, changing to an alternative database system might be the only option.

There are several ways to migrate the database from Microsoft Access to another database system. For example to do a database migration to PostgreSQL, there is a plugin for the database system called *pgadmin*. The *pgadmin* has a database migration wizard. There are also some other methods to do the migrations, such as using PERL scripts or using a third party application.

Implement Secure Connection. Currently, all of the communication between the user and the server uses plain Hyper Text Transfer Protocol (HTTP). In order to make the communication more secure, Secure Socket Layer (SSL) can be implemented.

Webopedia.com defines SSL:

> Secure Sockets Layer, a protocol developed by Netscape for transmitting private documents via the Internet. SSL works by using a public key to encrypt data that's transferred over the SSL connection. Both Netscape Navigator and Internet Explorer support SSL, and many web sites use the protocol to obtain confidential

user information, such as credit card numbers. By convention, URLs that require an SSL connection start with *https:* instead of *http*:.[7]

It is impossible to create a completely secure system. SSL provides security in terms of making the attempt on an unauthorized access to the system requires some extra amount of resources, such as processing power and time. The resources needed to access the system should be great enough so it will diminish the value of accessing the protected information.

The survey database contains some sensitive data. By accessing the database, someone can find out the names of graduate who work for a certain company or access the survey results for a certain graduate. By protecting the database itself and the connection to and from the database with a security protocol, personal information can be securely contained.

---

[7] http://www.webopedia.com/TERM/S/SSL.html

Chapter 5 Discussion and Conclusion

5.1 Discussion

The choice to use Microsoft Access was based on the requirement from the Assessment Office. Microsoft Access limitations might affect the performance of the survey in the future. Without the requirement of using Microsoft Access, the choice of the database system used for the project would be different.

It is possible to use Oracle as the database. The database design would not change but Oracle provides more advanced tools on the database side. For example: by using SQL*Plus, creating tables will be a simple task to do multiple times since the SQL scripts can be saved in a text file and can be re-used.

It is also possible to design the analysis tools within Oracle and eliminate the use of different applications to attain the analysis results of the survey. The analysis results can be obtained instantly based on the submitted surveys, since it is part of the system.

This project introduces this survey through the Internet for the first time. Currently, there only few survey results stored in the database. Trend analysis of demand assessment is not possible if based only on the current stored surveys. In the future, the accumulated data can be use to derive information either with the online analysis process or the data mining.

Another option for the database system is PostgreSQL. It has similar advance tools as in Oracle and it is free. This database system has some limitations. According to www.us.postgresql.org, PostgreSQL's limitations are:

Maximum size for a database	unlimited (60GB databases exist)
Maximum size for a table	64 TB on all operating systems
Maximum size for a row	unlimited in 7.1 and later
Maximum size for a field	1GB in 7.1 and later
Maximum number of rows in a table	Unlimited
Maximum number of columns in a table	1600
Maximum number of indexes on a table	Unlimited

Table 5.1 PostgreSQL's limitations

Oracle and PostgreSQL are well beyond Microsoft Access limitations. For a first time database user, Microsoft Access is a good first step. But for a huge repository system, a commercial database system should be used to maintain reliability.

5.2 Conclusion

For sure, making this report was not the easiest part of the project. It was a huge part of the project and you had no class to learn how to do it. It was a team effort, especially if you were not used to writing reports. Help in proofreading the report was really appreciated.

The project itself was a big compilation of time consuming tasks. Planning the project helped me to determine what to expect, but I still had surprises. Planning a project will get better as the planner gains more experience on doing the task.

Some of the tasks were on or even ahead of the schedule: design and create the form and design and create the database. Some other tasks were behind schedule. Most of them are tasks for the back end that support the system: database queries, page flows, error detections, etc.

It took several lines of codes to achieve a simple feature in this project. The project ends up with 23 files, 4 tables, and more than 2000 lines of HTML/ASP/JavaScript/CSS codes. It took a lot longer to finish the project than I have expected, but I have learned a lot from it.

Lessons from the graduate courses are not enough to meet all the project's requirements. Other resources are needed to fill the blanks and to go beyond the formally learned lessons.

The employer survey is up and running, waiting for the employers to fill in the survey forms. So far there have been some minor corrections. The problems were discovered by users and have been fixed. The survey is also open for more advanced improvement.

Microsoft Access can handle the work load for now. In the future, especially if the survey becomes a part of a huge relational database system, that has to be changed. That also will overcome the Microsoft Access limitations.

The extended architecture of the project with the addition of a database server can be seen on this following simple diagram:

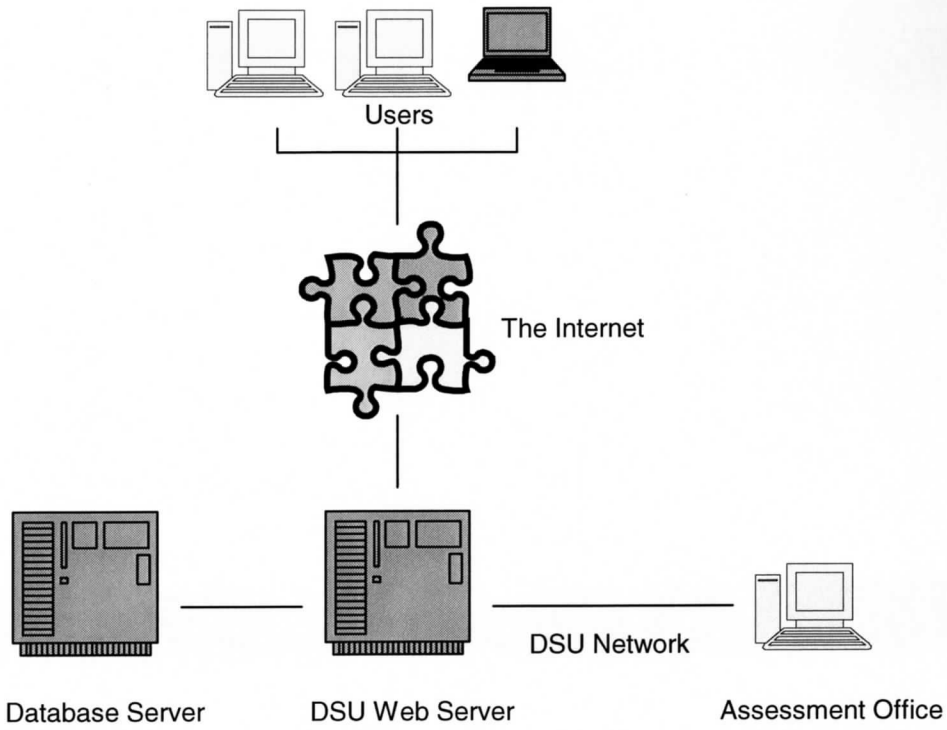


Figure 5.1 Advance system architecture for the project

On this opportunity, I would like to thank the committee members for their guidance and suggestions for the project and also for the report. I would like to thank Carrie Ahern and Marilyn Van Driesen for their help in proofreading the report and also thank Mr. Tom Farrell, Dr. Zehai Zhou, and Dr. Ronghua Shan for their suggestions and advice. I also like to thank Terry and Laurie Dennis for their guidance to make this project happened and also all the friends that I cannot mention their name one by one for their support and prayers through out the years.

References

American Psychological Association. (2001). *Publication Manual of the American Psychological Association* (5th ed.). Washington, DC: American Psychological Association.

Buser, D., Kaurffman, J., Llibre, J. T., Francis, B., Sussman, D., Ullman, C., et al. (2000). *Beginning Active Server Pages 3.0.* Birmingham, UK: Wrox Press Ltd. Printed in the United States.

Caterinicchia, D. (2001, September 3). Federal Computer Week. *DOE Lab Gets Results with Web-based Survey.* **Retrieved July, 2002, from http://www.fcw.com/fcw/articles/2001/0903/tec-doe-09-03-01.asp**

Clearform Software. *Microsoft Access Database Design.* **Retrieved August 24, 2002 from http://www.clearform.com/microsoft_access.htm**

McFadden, F. R., Hoffer, J. A., Prescott, M. B. (1999). *Modern Database Management* (5th ed.). United States: Addison-Wesley Educational Publishers, Inc.

WebSurveyor Corporation. *Paper and Web Survey Methods Compared.* **Retrieved July, 2002 from http://www.websurveyor.com/learn_news0110_bp.asp**

Webopedia. *SSL.* **Retrieved September 2, 2002 from http://www.webopedia.com/TERM/S/SSL.html**

Google Internet search result for "database connectivity asp cursor type". Retrieved July, 2002, from **http://www.devguru.com**

Google Internet search result for "javascript textarea form validation". **Retrieved April, 2002, from http://html.miningco.com/library/weekly/bl_aa042202-form2.htm**

Google Internet search result for "javascript radio button form validation". **Retrieved April, 2002, from http://www.w3schools.com/js/tryit.asp?filename=tryjs_form_radio**

Site Using PHP and MySQL. **Retrieved August 23, 2002 from http://www.mysql.com/articles/ddws/**

Appendices

Appendix A

Employer Survey Paper-based Format

Dakota State University
2002 Employer Survey

Name of graduate: **Title:** _____

Survey completed by: ___**Immediate Supervisor** ___**Personnel Manager**
___**Other**_____

I. General university questions

Please rate the DSU graduate in the following areas:

	Very Good				**Very Poor**	**Not applicable/ Don't know**
Computer Skills:						
1. Use of computer software (e.g. word processing, spreadsheets)	5	4	3	2	1	0
2. Ability to adapt to changes in computing environment	5	4	3	2	1	0
3. Overall computer knowledge (e.g. hardware and software)	5	4	3	2	1	0
Communication/Socialization Skills:						
4. Written communication skills	5	4	3	2	1	0
5. Oral communication skills	5	4	3	2	1	0
6. Appreciates the cultural and ethnic differences among people	5	4	3	2	1	0
7. Interpersonal skills (e.g. working in a ream setting)	5	4	3	2	1	0
Job-Related Skills:						
8. Ability to solve work-related problems	5	4	3	2	1	0
9. Ability to find, evaluate and apply information	5	4	3	2	1	0
10. Ability to use information ethically	5	4	3	2	1	0
11. Knowledge of academic area as it relates to his/her position	5	4	3	2	1	0
12. Ability to learn on the job	5	4	3	2	1	0
13. **Overall satisfaction with employee's job-related skills**	5	4	3	2	1	0

14. To what extend dos this employee use computers in his/her work?
a. Daily b. Several times per week c. At least once per week d. At least once per month e. Never

15. At what level did this employee require additional training in **computer skills**?
a. Significantly less than usual b. Less than usual c. Usual d. More than usual e. Significantly more than usual

16. At what level did this employee require additional training in **communication skills**?
a. Significantly less than usual b. Less than usual c. Usual d. More than usual e. Significantly more than usual

17. At what level did this employee require additional training in **job-related skills**?
a. Significantly less than usual b. Less than usual c. Usual d. More than usual e. Significantly more than usual

18. At what level did this employee require additional training in **interpersonal skills**?
a. Significantly less than usual b. Less than usual c. Usual d. More than usual e. Significantly more than usual
19. What on the job training is being delivered to graduates after they are hired that we could be providing before they start their new job?

20. What job-related skills are most essential for your new employees?

Your comments are appreciated!
Comments regarding ratings in **computer skills**:

Comments regarding ratings in **communications skills**:

Comments regarding ratings in **job-related skills**:

Comments regarding ratings in **interpersonal skills**:

Other comments:

Dakota State University
2002 Employer Survey
Education Graduates Section

Please rate the education graduate's ability to:

	Exceeds Expectation		Meets Expectation		Needs Improvement	Not Applicable
1. Demonstrate an understanding of the central concepts, tools of inquiry, and structures of the content/subject matter/disciplines s/he teaches.	5	4	3	2	1	0
2. Create learning experiences that make the content/subject matter/disciplines meaningful for learners.	5	4	3	2	1	0
3. Demonstrate an understanding of how students learn, construct knowledge, and how students develop.	5	4	3	2	1	0
4. Provide learning opportunities that support the intellectual, social, and personal development of students.	5	4	3	2	1	0
5. Demonstrate an understanding of how learners differ in their approaches to learning and the ability to create instructional opportunities that are adapted to diverse learners.	5	4	3	2	1	0
6. Use a variety of instructional strategies to encourage learners' development of critical thinking, problem solving, and performance skills.	5	4	3	2	1	0
7. Use an understanding of individual and group motivation and behavior to create a learning environment that encourages positive social interaction, active engagement in learning, and self-motivation.	5	4	3	2	1	0
8. Use knowledge of effective verbal, nonverbal, and media communication techniques to foster active inquiry, collaboration, and supportive interaction in the classroom.	5	4	3	2	1	0
9. Plan instruction based upon knowledge of subject matter, learners, the community, and curriculum goals.	5	4	3	2	1	0
10. Understand and use formal and informal assessment strategies to evaluate and ensure the continuous intellectual, social, and physical development of the learner.	5	4	3	2	1	0
11. Reflect on and evaluate the effects of their choices and action on others (students, parents, and other professionals in the learning community).	5	4	3	2	1	0
12. Seek opportunities for professional growth and responsibility.	5	4	3	2	1	0
13. Collaborate with school colleagues, parents, and agencies in the larger community to support students' learning and well-being.	5	4	3	2	1	0
14. Demonstrate an understanding of the capabilities of technology and its impact on education.	5	4	3	2	1	0
15. Integrate technology into the teaching and learning process.	5	4	3	2	1	0

16. Demonstrate appropriate moral, legal, ethical, and political judgments in professional decisions. 5 4 3 2 1 0

Please respond to these following questions:

17. Of the knowledge, skills, or attitudes referenced above, in what areas was the graduate **MOST** prepared?

18. Of the knowledge, skills, or attitudes referenced above, in what areas was the graduate **LEAST** prepared?

19. Based on your observations of the graduate's performance, please make suggestions for improvement of the DSU teacher education program

Appendix B

Project Proposal

Project Proposal

Online Survey for Assessment Office

Employer Survey and Graduate Survey

Introduction

This project is part of Graduate Assistantship work for the Assessment Office. By making the survey accessible online, it is expected that the results will be more accurate and the response time will be faster.

Carrie Ahern requested that both surveys be converted from the traditional paper-pencil format to a web-based format. The results will be more accurate since the critical information about surveyed subjects (i.e student's name, surveyor position.) will be provided as options; user input mostly will be mouse clicks on the options.

History

The DSU Employer Survey has been used since 1993 to survey employers of the previous year's graduates. The employers rate the DSU graduates in three areas: computer skills, communication/socialization skills and job-related skills. Additional questions on the survey relate to frequency of computer use and level of training. This information is used as one method of determining if the goals and outcomes of DSU's general education and major-field assessment plans are being met.

The DSU graduate survey, sent to graduates one and three years after graduation, has been used since the early 1980s to assess graduates' level of satisfaction in selected areas. The questions on the survey relate to

the graduates' satisfaction with their competence in three areas: computer skills, communication/socialization skills and job-related skills. The graduates also indicate how much DSU contributed to these competencies. The information is used as one method of determining if the goals and outcomes for the general education and major-field assessment plans are being met.

The DSU Employer and Graduate Survey reports are distributed to the Assessment Coordinating Committee. Summaries of the reports are included on the assessment website. Each college receives a report containing information specific to their majors.

Current Process

The current process consists of:

- Coding surveys with a sequential number linked to an Access database
- Sending copies of the surveys via US mail
    - A formal letter is included which contains assurance of confidentiality and a notice that the summary results will be posted on the web
    - A return envelope is included
- Survey data is entered into Access
- To generate the survey report, the survey database will be exported to an Excel table. SAS is used to calculate statistical information. Survey reports are made based on this statistical information.

The Project

The surveys will reside on DSU web server. ASP and HTML will be used to make the pages. The survey results will be stored directly to Access database. By providing these surveys online, it will eliminate of the process of typing in data into Access. It also makes the surveys more accessible and less time consuming than the paper-based ones.

The surveys can be delivered by hyperlink through email. This is more efficient than the traditional process of completing the paper-pencil version and returning by US mail. The surveys will be available 24 hours a day 7 days a week, as long the web server is available. There is also no risk of the surveyor misplacing the survey.

Verification can be done since all submitted answers will have a time stamp and the IP address of the computer used to fill the survey. The intention of this project is to make the surveys more reliable, less time consuming and more efficient.

Skill and Knowledge

In order to make the project possible, it needs good planning. Project design is also necessary. The first step in doing this project is to design a good database. The database will be a simple design since the requirement doesn't need a complicated database system.

The WBS and Gantt chart will help to control and evaluate the ongoing project. By comparing the project to WBS and Gantt chart, one can restrict the project so it will minimize the risk of scope creep. We will also know whether the project is ahead or behind the scheduled time.

Since the project has a repository system, knowledge about database is required. Introduction to database knowledge should be sufficient. Server side will be selected for this project to give compatible feature. The user can use any browser to complete the survey. In order to conform to the current infrastructure, combination of ASP and HTML will be use to create the page. The survey page will be connected to Access through data store link driver in ASP.

Work Breakdown Structure

Research

- HTML Coding references
- ASP Coding references
- Access Database references

Get requirements

- Data types
- Primary keys
- Valid values
- Survey Questions

Design

- Database relationship
- Data store connection

- Survey form

Create web pages and database

- HTML and ASP coding
- Database tables
- Database meta data

Test and Evaluate

- Debugging
- Error messages
- ASP page
- Database table
- Data validation

Implement the project

- Form hyperlinks
- Assessment office evaluation

Conclusion

The new process for distributing, collecting and analyzing the Employer and Graduate surveys will be more efficient and less time consuming. In addition, the web-based survey format aligns with the technology-based mission of DSU.

Appendix C Gantt Chart